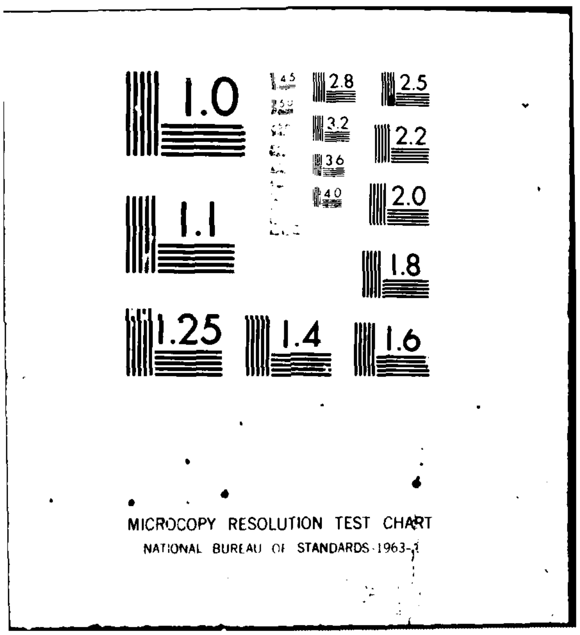
AD-A084 892 PARKER (LEE W) INC CONCORD MASS F/G 9/2
TIME-DEPENDENT COMPUTER MODEL OF PLASMA SPACE CHARGE INTERACTIO--ETC(U)
DEC 79 L W PARKER F19628-78-C-0149
UNCLASSIFIED AFGL-TR-80-0018 NL

1 OF 1
AD
A084 892

END
DATE
FILMED
6-80
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFGL-TR-80-0018

LEVEL II

12

ADA084892

TIME-DEPENDENT COMPUTER MODEL OF PLASMA SPACE CHARGE INTERACTIONS WITH A FINITE-CYLINDRICAL SPACECRAFT

Lee W. Parker

Lee W. Parker, Inc.
252 Lexington Road
Concord, Massachusetts 01742

DTIC ELECTE MAY 30 1980 D

31 December 1979

Final Report
1 April 78 - 30 September 79

AIR FORCE GEOPHYSICS LABORATORY
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
HANSCOM AFB, MASSACHUSETTS 01731

80 5 27 291

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFGL-TR-80-0018
2. GOVT ACCESSION NO.: AD-A084 892
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): TIME-DEPENDENT COMPUTER MODEL OF PLASMA SPACE CHARGE INTERACTIONS WITH A FINITE-CYLINDRICAL SPACECRAFT,
5. TYPE OF REPORT & PERIOD COVERED: Final Rep[redacted]t, 1 Apr[redacted] 78 - 30 Sep[redacted]79
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Lee W. Parker
8. CONTRACT OR GRANT NUMBER(s): F19628-78-C-0149
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Lee W. Parker, Inc. 252 Lexington Road Concord, Massachusetts 01742
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62101F 766107AE
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Geophysics Laboratory Hanscom AFB, Massachusetts 01731 Monitor/Allen G. Rubin/PHG
12. REPORT DATE: 31 Dec[redacted]79
13. NUMBER OF PAGES: 85
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report):
18. SUPPLEMENTARY NOTES:
19. KEY WORDS (Continue on reverse side if necessary and identify by block number):
   - computer model
   - spacecraft charging
   - time dependent
   - cylindrical spacecraft
   - space charge

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer program has been developed for the study of the time-dependent behavior of the sheaths and charging effects of spacecraft with finite-cylinder ("pillbox") geometry. The behavior of ions and electrons is simulated by following representative computer particles in time. During each time step the particles are moved, their space charge is computed, and the electric potential distribution is updated. The development is original. A number of sample runs show that current collection and space charge behavior can be computed as functions of time.

DD FORM 1 JAN 73 1473

Unclassified 391800

INTRODUCTION

This project is concerned with the numberical simulation of the time-dependent behavior of spacecraft sheaths and charging effects. The objectives are to develop a time-dependent simulation model for plasma-spacecraft interactions, and to apply this model to a geometrical representation of the SCATHA spacecraft. An appropriate representation is that of a short right-circular cylinder (or "pillbox"), of comparable length and diameter (see Fig. 1), with azimuthal symmetry so that the potential and charge distributions can be defined on a grid in r-z space.

A computer program has been developed for the study of the time-dependent sheath. A grid is used to define the spatial distributions of potential and charge density in the space around the spacecraft. Figure 2 illustrates the nature of the r-z grid representation used. The geometry is axially-symmetric, with the axis shown as the vertical dotted boundary line on the left, labelled "West". The boundary condition representing the condition on the potential at infinity is applied to the other boundary lines of the grid. The inner boundary represents the satellite surface, on the grid points of which the surface potentials are defined. Associated with each spatial grid point is a volume of revolution in the shape of a torus of rectangular cross-section (shown as shaded boxes surrounding some of the grid points). Only 24 grid points are shown in Fig. 2, for the purpose of clarity. In an actual problem many more points are used.

An important feature of this grid representation is that the zoning is nonuniform. This allows for fine zoning in the regions of interest, e.g., where there are large gradients, and coarse zoning elsewhere, and has the advantage of optimum use of a given number of grid points, and therefore computer efficiency in large problems.

The plasma electrons and ions are simulated by a number of discrete "computer particles" injected through the outer grid boundaries. These particle trajectories are followed step by step through the grid. Similarly, emitted electrons and/or ions are injected from the inner (spacecraft) surfact. Each simulation particle represents a definite number of real particles, retaining the charge-to-mass ratio of the real particles. The density

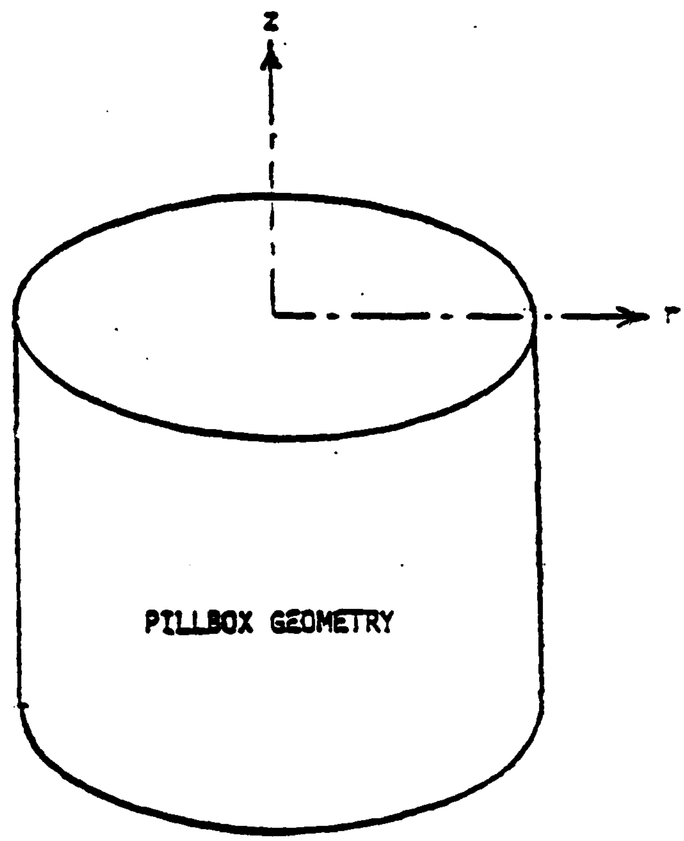


FIG 1. SPACECRAFT GEOMETRY

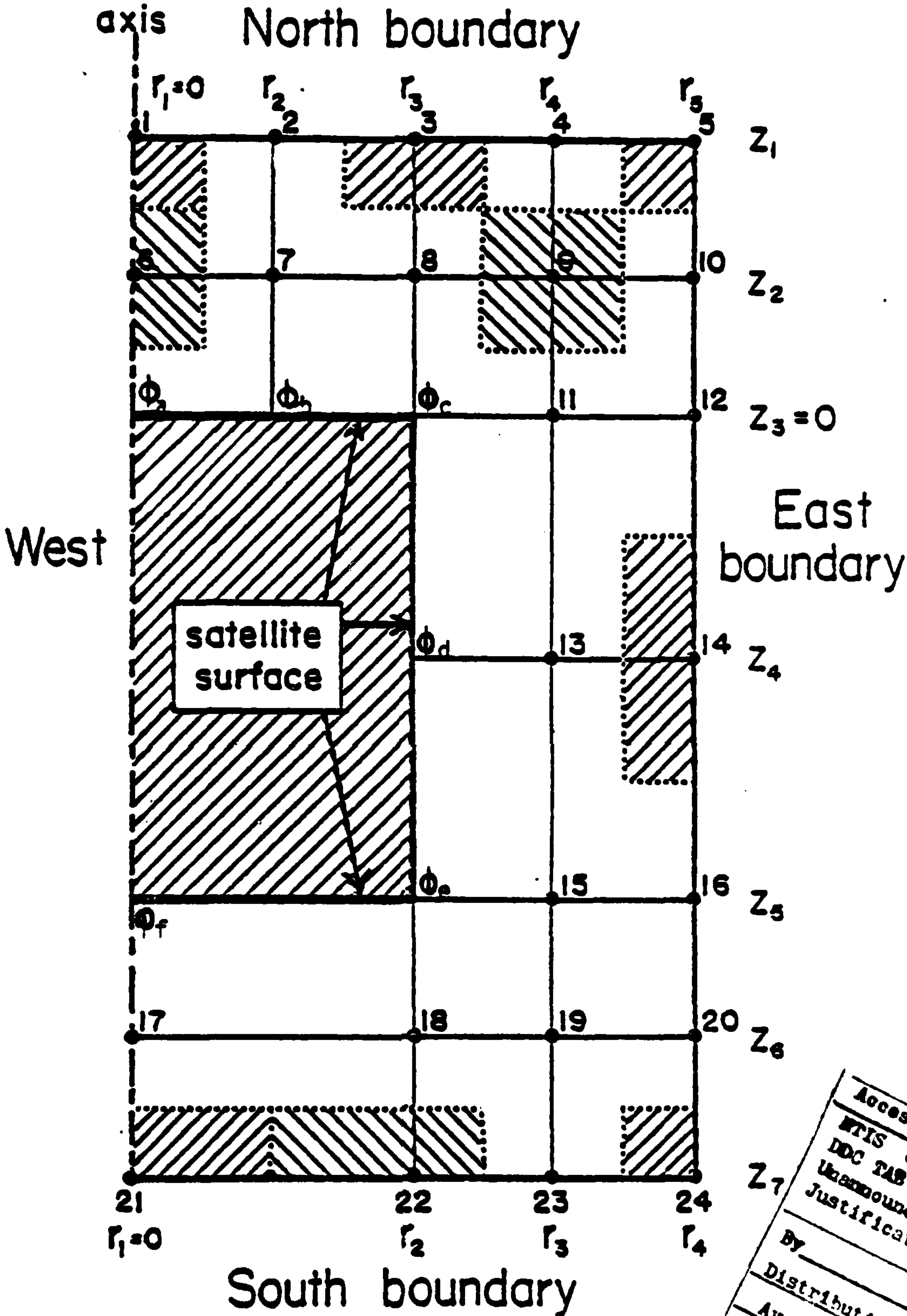


FIG. 2. GRID REPRESENTATION

of computer particles associated with any grid point is determined at any instant of time by the number of these particles in the box surrounding that grid point at that time. The potential distribution is found for the particular charge distribution and boundary conditions at any given time by solving the discretized analog of the Poisson equation. The particles are "pushed" in the calculated electric field during a time step in a given cycle and, based on their positions at the end of the time-step new charge densities are assigned to the grid points. At the beginning of the next cycle a new potential distribution is calculated from these densities. The calculation thus simulates the time-behavior of the system step by step.

In the following sections we describe the particle injection algorithms, trajectory calculation method, and technique for solving the Poisson equation. Following this we discuss the results of a number of sample runs. These show that the code is capable of handling arbitrary densities and temperatures with acceptable fluctuations. Current collection and space charge distributions can be computed as functions of time. Finally the program listing of PARKTDC (Parker Time Dependent Charging) is presented.

The present program is an original development. Although there is a considerable numerical simulation literature, it applies to other geometries than the present one. Sources available to the author give little or no information on methods of simulating axially-symmetric time-dependent problems involving finite cylinders in space plasmas. Yet the axially-symmetric 3-dimensional geometry requires techniques significantly different from those of say, planar, infinitely-long cylinder, or cartesian systems. Therefore no references are cited.

PARTICLE INJECTION ALGORITHMS

Particles are injected through the outer grid boundary according to the following velocity-distribution options: (a) unidirectional monoenergetic beam, (b) isotropic monoenergetic, and (c) isotropic Maxwellian.

Selection from isotropic monoenergetic velocity distribution:

The outer boundary surface of the grid is a cylindrical "box" of finite length H and radius R_o. This means that in the process of statistical selection of particle injection sites, one of the three areas must be selected, namely the top (A), side (B), or bottom (C). Consider a distribution of particle beams distributed in polar angle θ (with respect to the cylinder axis), at a fixed energy. For an isotropic angle distribution, choose θ from $\cos\theta$ uniformly distributed in the range (-1, 1). Define the projections of the cylinder areas onto a plane perpendicular to the beam. There are two cases:

Case $\theta > \pi/2$

Define $A = \pi R_o^2 |\cos\theta|$ and $B = 2R_o H\sin\theta$, and choose random number to select A or B.

Case $\theta < \pi/2$

Define $C = \pi R_o^2 \cos\theta$ and $B = 2R_o H\sin\theta$, and choose random number to select B or C.

With angle θ and areas selected:

If the selected surface is A or C, choose azimuthal angle ϕ uniformly from (o,π), and choose r^2 uniformly from $(0,R_o^2)$. This gives the position of injection on the surface. If the surface selected is B, then choose z uniformly from (0,H), and choose h uniformly from $(0,R_o)$. Then obtain ϕ from arc $\sin(h/R_o)$.

It should be noted that alternatively one could have first selected the areas A and B (or A and C) on the basis of a uniform distribution of values of the integral

$$\frac{1}{\pi}\int_o^\pi \frac{\sin\theta\,|\cos\theta|\,d\theta}{|\cos\theta| + (2H/\pi R_o)\sin\theta} \qquad (1)$$

which represents the average over angles θ of the ratio of A to A+B, but which does not seem to be evaluable in analytical terms. Following this selection one would then select a value for θ. This method would be equivalent to the one used.

Components of velocity at injection point:

For a given speed v, and for selected angles θ and φ, we have the components of particle velocity at the point of injection:

$$\begin{aligned} \dot{x} &= v \sin\theta \cos\phi \\ \dot{y} &= v \sin\theta \sin\phi \\ \dot{z} &= v \cos\theta \end{aligned} \tag{2}$$

Selection from Maxwellian velocity distribution:

Let the velocity distribution $f(\vec{v})$ be described by

$$f(\vec{v})d^3\vec{v} \sim e^{-v_\perp^2} v_\perp dv_\perp \cdot e^{-v_z^2} dv_z \tag{3}$$

in terms of dimensionless velocity components, where v_z is the axial component of velocity, and $v_\perp$ is the perpendicular component $\sqrt{(\dot{x}^2 + \dot{y}^2)}$. Then choose $v_\perp$ from $RAN_1 = \exp(-v_\perp^2)$, and v_z from $RAN_2 = \mathrm{erf}(v_z)$, where RAN_1 denotes a random number uniformly distributed in the unit interval, while RAN_2 denotes a random number uniformly distributed in the range (-1,1). For v_z we need the inverse ($=\mathrm{erf}^{-1}$) of the error function erf. (The inverse function was constructed by suitably modifying a fitting formula given by Abramowitz and Stegun.) Then the angle θ and speed v are given by $\tan\theta = v_\perp/v_z$, and $v = \sqrt{v_\perp^2 + v_z^2}$. Having the angle θ, we then select the injection point as given earlier, and following this the velocity components $\dot{x}$, $\dot{y}$, and $\dot{z}$. If there is also a drift velocity M (= Mach number) in the axial direction, then v_z can be chosen from $RAN_2 = \mathrm{erf}(v_z - M)$. This method of selection thus leads to a simplification when dealing with a drifting Maxwellian distribution.

PARTICLES INJECTED IN A TIME STEP

The actual charge of particles entering a surface of area A in time Δt is $ej_0 A\Delta t$, where j_0 is the random thermal flux of incident particles

and e is particle charge. Let j_0 be given by $n_0 v_\perp$, where n_0 is the plasma density, and $v_\perp$ is the mean velocity component of incident real particles normal to the surface. For a choice of time interval $\Delta t = \Delta z / v_\perp$, where Δz is a chosen thickness of transit (say, of the order of a mesh interval), we may determine the number N of real particles injected during the time step:

$$N = j_0 A \Delta t = n_0 A \Delta z \tag{4}$$

If N' is the chosen number of computer particles injected in a time step, say 10 to 1000, the charge e' per computer particle is given by

$$e' = \frac{Ne}{N'} = \frac{e n_0 A \Delta z}{N'} \tag{5}$$

In the problem of isotropic injection, A may be evaluated as $2\pi R_0^2 (1 + H/R_0)$. The value of e' is used to calculate the charge assigned to a grid point through its product with the number of computer particles found at a given instant of time to be associated with that grid point. It is similarly used to compute the charge incident on and absorbed by the body.

Typical values of the parameters which have been used in tests are the following:

$R_0 = 100$ cm $\qquad N' = 10$ to 1000

$H = 100$ cm

$\Delta z = 2$ cm

$n_0 = 10^3/\mathrm{cm}^3$ to $10^5/\mathrm{cm}^3$

For $n_0 = 10^3$, $N = 2.51 \times 10^8$ is the number of real particles injected per time step. Each computer particle then represents N/N' real particles, or 2.51×10^5 to 2.51×10^7.

TRAJECTORY CALCULATION

The particles are moved in accord with the equations of motion, which may be represented as follows.

Let X denote the dimensional vector position (X stands for any one of the 3 cartesian coordinates). Let T denote the dimensional time, and let V denote the dimensional potential (in volts) at the position X. Then over a short step ΔT with constant acceleration, X changes in accord with

$$X = X_o + \dot{X}_o \Delta T - \frac{q}{2m} \frac{dV}{dX} (\Delta T)^2 \tag{6}$$

where q and m are the particle charge and mass, and where X_o is the initial value of X at the beginning of the time step. Let dV/dX be in volts/cm, and let ΔT be expressed as

$$\Delta T = \frac{(\Delta Z)_{min}\ \Delta t}{v_{oi}} \tag{7}$$

where $(\Delta Z)_{min}$ is the minimum zone thickness, v_{oi} is the scale velocity of the ions, and Δt is a dimensionless time, called "DELTA" in the program. Let m be the mass of the particle expressed in units of the ion mass. Then we may write

$$X = X_o + \dot{X}_o \frac{(\Delta Z)_{min}\ \Delta t}{v_{oi}} - \frac{(\Delta Z)^2_{min}\ (\Delta t)^2}{4(m/m_i)} \frac{d}{dX} \left(\frac{qV}{E_i}\right) \tag{8}$$

where E_i(ev) is the scale energy of the ions ($= m_i v_{oi}^2/2$) called "TVIONS" in the program, and qV is the potential energy in ev. Similarly we obtain the velocity components:

$$\dot{X} = \dot{X}_o - v_{oi} \frac{(\Delta Z)_{min}\ (\Delta t)}{2(m/m_i)} \frac{d}{dX} \left(\frac{qV}{E_i}\right) \tag{9}$$

Hence, assuming electrons and one species of ion, the program

(a) reads in:

Δt = "DELTA"

E_i(ev) = "TVIONS" = ion energy

E_e(ev) = "TVELEC" = electron energy

m_i/m_e = "XMASS" = mass ratio

and

(b) constructs the scale velocity

$$v_o = \text{"SPEED"} = 5.93 \times 10^7 \sqrt{\text{TVIONS/XMASS}}$$

$$= \text{ion speed in cm/sec}$$

and

(c) finds the minimum zone thickness $(\Delta Z)_{min}$.

The particles are moved in subroutine "TRACK" (called by "DENSTY"), with given

$$X,\ \dot{X}/\text{SPEED, and "DT"} = (\Delta Z)_{min} \times \text{DELTA}$$

With the potential grid replaced by $\phi=(m_i/m)qV/E_i$, the new values of X and $\dot{X}$ are given by:

$$X = X_o + \dot{X}_o \frac{DT}{v_o} - \frac{(DT)^2}{4} \frac{d\phi}{dX} \tag{10}$$

$$\dot{X} = \dot{X}_o - v_o \frac{DT}{2} \frac{d\phi}{dX} \tag{11}$$

<u>Interpolation</u>

The required components of $d\phi/dX$ in the latter equations are obtained by double linear interpolation within the boxes of the grid. Let r and z be located in the ranges $r_j \le r < r_{j+1}$ and $z_i \le z \le z_{i-1}$. Then the interpolated values of $\partial\phi/\partial r$ and $\partial\phi/\partial z$ are given by:

$$\frac{\partial\phi}{\partial r} = [\phi(i,j+1) - \phi(i,j) + (z-z_i)G/D_z]/D_r \tag{12}$$

$$\frac{\partial\phi}{\partial z} = [\phi(i-1,j) - \phi(i,j) + (r-r_j)G/D_r]D_z \tag{13}$$

where

$$G = \phi(i-1,j+1) + \phi(i,j) - \phi(i-1,j) - \phi(i,j+1) \tag{14}$$

and

$$D_r = r_{j+1} - r_j,\ D_z = z_{i-1} - z_i \tag{15}$$

The interpolated potential itself is given by

$$\begin{aligned} \phi = \phi(i,j) &+ (r-r_j)[\phi(i,j+1) - \phi(i,j)]/D_r \\ &+ (z-z_i)[\phi(i-1,j) - \phi(i,j)]/D_z \\ &+ (r-r_j)(z-z_i)G/D_rD_z \end{aligned} \tag{16}$$

THE POISSON PROBLEM: POISSON DIFFERENCE EQUATIONS

In the present problem the electrostatic field is axially symmetric and is defined on a mesh of spatial grid points, such that at any point (including grid points) the potential and electric field can be obtained by interpolation.

Assume that the space charge density is known at the grid points. Consider a group of interior grid points, forming a portion of the overall grid as shown in Fig. 3. In this figure, the vertical and horizontal directions are the z and r directions, respectively, where z and r denote the cylindrical axial and cylindrical radial coordinates, respectively. Three horizontal grid lines, of constant z-values z_{i-1}, z_i, and z_{i+1}, and three vertical grid lines, of constant r values r_{j-1}, r_j, and r_{j+1}, are shown in the figure. (Note that the index (i) of z increases as z decreases.) The set of grid lines intersect at 9 grid points, or nodes, as shown. Each point may be considered to be associated with a volume of space, and to have a group of four neighboring points which "interact" with it. Thus, consider the central point of the group, labeled C in the figure, which may be identified with one of the grid points in Fig. 2. Associated with this point is a volume of revolution (a torus) whose cross-section is rectangular and is shown by the rectangular shaded area surrounding Point C. The shaded area is defined by connecting the mid-points of the surrounding mesh rectangles. Let τ denote the volume of the torus, and let the neighboring points (above, below, to the right of, and to the left of C) be labeled N, S, E and W (north, south, east and west, respectively).

Let the Poisson equation be written

$$\nabla^2\phi = -\rho/\varepsilon \qquad (17)$$

where ρ denotes the space charge density, and ε denotes the dielectric constant.

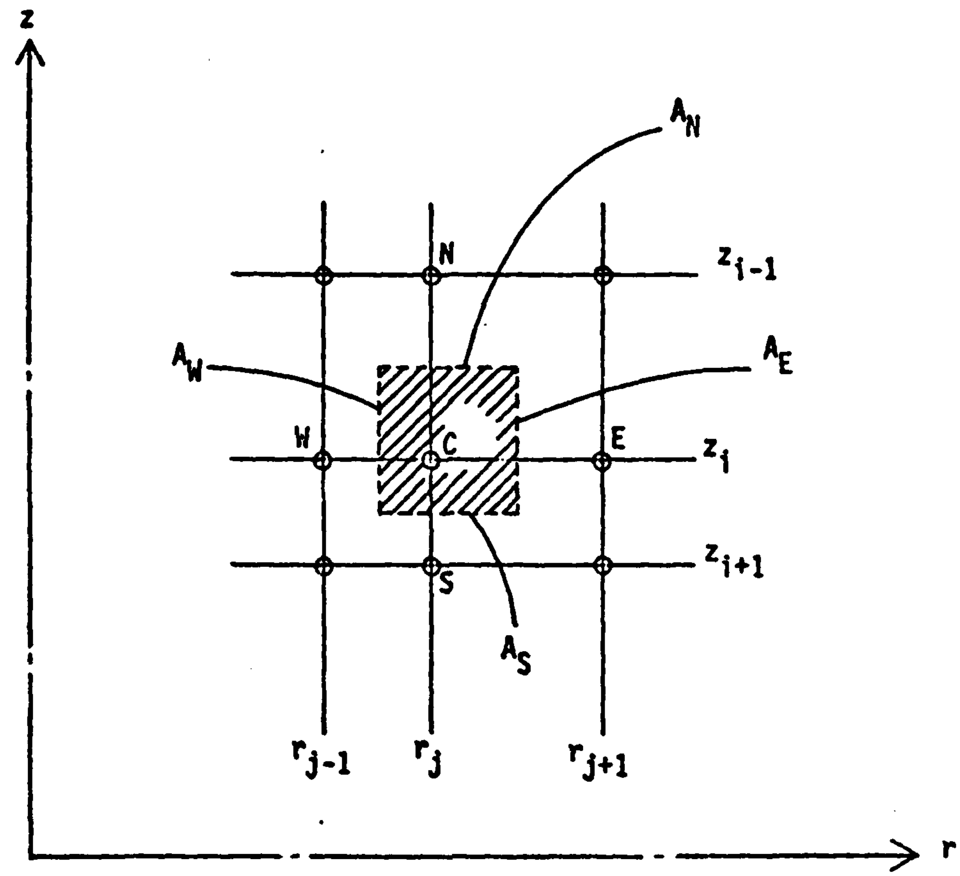


FIG. 3. GROUP OF INTERIOR GRID POINTS IN r,z GRID

The grid lines may be considered to be arbitrarily chosen so that the mesh intervals are nonuniform. In this case the Poisson difference equations may be obtained by integrating Eq. (17) over the volume τ of the torus associated with Point C:

$$\iiint_\tau \nabla^2\phi \, d\tau = - \iiint_\tau \rho d\tau \cong - Q_C/\varepsilon \tag{18}$$

where Q_C is known at the grid point C. The right-hand side has been approximated as shown since τ is small in principle, and Q_C is the net charge in the box at Point C. By the divergence theorem, the left-hand side becomes

$$\iint_\Sigma \frac{\partial\phi}{\partial n} d\Sigma \cong$$

$$A_N \left(\frac{\partial\phi}{\partial n}\right)_N + A_S \left(\frac{\partial\phi}{\partial n}\right)_S + A_E \left(\frac{\partial\phi}{\partial n}\right)_E + A_W \left(\frac{\partial\phi}{\partial n}\right)_W \tag{19}$$

where Σ denotes the surface of the torus; $\partial\phi/\partial n$ is the component of $\nabla\phi$ in the outward normal direction at the surface; A_N, A_S, A_E, and A_W denote the areas of the north, south, east, and west surfaces, respectively; and the quantities $(\partial\phi/\partial n)_{N,S,E,W}$ denote values of $\partial\phi/\partial n$ taken to be constant on the corresponding surfaces.

$(\partial\phi/\partial n)_{N,S,E,W}$ may be approximated by difference quotients, namely,

$$\left(\frac{\partial\phi}{\partial n}\right)_N \cong \frac{(\phi_N - \phi)}{(z_{i-1} - z_i)} \qquad \left(\frac{\partial\phi}{\partial n}\right)_S \cong \frac{(\phi_S - \phi)}{(z_i - z_{i+1})}$$

$$\left(\frac{\partial\phi}{\partial n}\right)_E \cong \frac{(\phi_E - \phi)}{(r_{j+1} - r_j)} \qquad \left(\frac{\partial\phi}{\partial n}\right)_W \cong \frac{(\phi_W - \phi)}{(r_j - r_{j-1})} \tag{20}$$

where ϕ denotes the potential at Point C and ϕ_N, ϕ_S, ϕ_E, ϕ_W denote the neighboring potentials. If Point C is an interior grid point, the areas A_N, A_S, A_E, and A_W are given by

$$
\begin{aligned}
A_N &= \frac{\pi}{4}\left[(r_{j+1} + r_j)^2 - (r_j + r_{j-1})^2\right] \\
A_S &= A_N \\
A_E &= \frac{\pi}{2}(r_{j+1} + r_j)(z_{i-1} - z_{i+1}) \\
A_W &= \frac{\pi}{2}(r_j + r_{j-1})(z_{i-1} - z_{i+1})
\end{aligned} \tag{21}
$$

and the volume τ is given by

$$\tau = \frac{A_N}{2}(z_{i-1} - z_{i+1}) \tag{22}$$

Thus we obtain the difference equation in the form

$$C_N \phi_N + C_S\phi_S + C_E\phi_E + C_W\phi_W - C\phi = - Q_c/\varepsilon \tag{23}$$

where

$$C = C_N + C_S + C_E + C_W \tag{24}$$

and

$$
\begin{aligned}
C_N &= \frac{A_N}{(z_{i-1} - z_i)} \qquad & C_S &= \frac{A_S}{(z_i - z_{i+1})} \\
C_E &= \frac{A_E}{(r_{j+1} - r_j)} \qquad & C_W &= \frac{A_W}{(r_j - r_{j-1})}
\end{aligned} \tag{25}
$$

This shows how to form the difference equations used for the Poisson problems of this report. Equation (24) holds only for an "interior" point of the grid, that is, a point surrounded by neighbors on all four sides. If Point C has a known neighboring potential (for example, if Point C is adjacent to the spacecraft surface), then the corresponding term on the left-hand side of Eq. (23) is transferred to the right-hand side as a known quantity.

The boundary conditions for the potentials in the Poisson problem are as follows. At points representing the body surface, the normalized potentials are fixed at the chosen values. At the external (boundary) points of the grid, where "infinity" is represented on the computer, a "floating" condition is optionally used, namely, a linear relation between ϕ and $\partial\phi/\partial n$, the normal component of $\nabla\phi$. The exact relation of ϕ to $\partial\phi/\partial n$ is not important when the external boundary of the grid is sufficiently far away. (For the calculations to be reported, the assumed relation was the same as for a Coulomb potential.) In any case, either the fixed condition $\phi = 0$ or the floating condition will give the same results, provided the grid boundary is moved sufficiently far out. The effects of various types of boundary conditions representing "infinity" have been studied by the author. In general, the floating condition appears to be computationally more efficient than the fixed one. Of course, the floating condition becomes ideal when the true relation between ϕ and $\partial\phi/\partial n$ is used, but this requires that the asymptotic form of the solution be known in advance. The boundary conditions at the outer grid surfaces can be combinations of fixed and floating conditions.

Consider a Point C on the outer boundary of the grid where a floating boundary condition is chosen. If the potential is assumed to satisfy the linear law

$$\frac{\partial\phi}{\partial n} = \frac{\partial\phi}{\partial z} = - \alpha\phi \tag{26}$$

on the z-boundary (North or South), and

$$\frac{\partial\phi}{\partial n} = \frac{\partial\phi}{\partial r} = - \beta\phi \tag{27}$$

on the r-boundary (East only; β=0 on the West), then the corresponding "neighbor term" on the left-hand side of Eq. (23) vanishes, and the corresponding "neighbor coefficient" on the right-hand side of Eq. (24) is replaced by αA or βA, where A is the appropriate area. The quantities α and β depend on the position and on the assumed model for the variation of the potential at large distances.

Once the coefficients of all of the equations (corresponding to the grid points where the potentials are unknown) are computed, the system of linear equations of the form of Eq. (23) may be solved by iteration. Point-successive over-relaxation is a well-known process and has been found to be effective in the present problem. For the relaxation process, one rearranges the equations, so that the "diagonal" term is alone on the left-hand side, while all the other terms are on the right-hand side with the known charge-density term. Thus, Eq. (23) becomes

$$C\phi = C_N\phi_N + C_S\phi_S + C_E\phi_E + C_W\phi_W + Q_c/\epsilon \tag{28}$$

First, an initial guess is made for the values of all the potentials. Then new values are obtained from the left-hand sides of all of the equations (28), using previous values on the right-hand sides. One "sweeps" through the equations successively, replacing the potentials on the right-hand sides with updated values as they become available from preceding equations. This procedure is usually stable and leads to convergence. "Over-relaxation" is the process of mixing successive potential iterates in such a way as to enhance the rate of convergence.

It is convenient to express all potentials in volts and all lengths in centimeters. Then if the charges are all expressed in picocoulombs, we need simply to replace $1/\epsilon$ by $0.9 \times 4\pi$ or 3.6π, in vacuum.

SAMPLE RESULTS

One of the most vexing problems of numerical time simulation is that of noise, wherein fluctuations not representative of the actual plasma are generated by the relatively small numbers of particles and grid points used. This problem becomes more severe as the plasma density increases, that is, as the Debye length becomes small compared with the spacecraft size. Since it was deemed essential that the code be able to cope with dense as well as tenuous plasmas, as part of the code checkout sample runs were made with the parameters, temperature T=0.1 ev, and density n_o varying from $10^3/cm^3$ to as high a limit as practicable.

Among the quantities of interest are the collected fluxes of ions and electrons. Fluctuations in these are a measure of those in the plasma. In the first problem to be discussed the spacecraft is modeled by a disk, of diameter one meter. The plasma has density $1000/cm^3$, and is Maxwellian with T=0.1 ev. In this case the Debye length is 7.43 cm, so that the Debye number is 0.149. The electron and ion plasma periods are 3.5×10^{-6} sec and 1.8×10^{-5} sec, respectively, for an assumed ion mass of 25 electron masses. (The use of an unphysically small ion mass is common in simulations, especially if the steady state is wanted. In the steady state the solution normally does not depend on the ion mass.)

The disk is assumed to "turn on" at t=0 with a fixed potential of -10 v. Time steps of length 2.36×10^{-6} sec are chosen, with 100 electrons and 20 ions injected per step. (The ratio 100 to 20 is the same as the ratio of electron-to-ion random thermal fluxes.) A coarse grid of 25 points is used.

The number of ions absorbed per step represents the collected ion current. This quantity is shown in Fig. 4 as a function of cycle (step) number. It is strongly fluctuating, but quite clearly centered about 12/step. Increasing the number injected per step to 1000 electrons and 200 ions results in a smaller fluctuation as illustrated in Fig. 5. When 10-cycle averages are plotted instead of the raw data, one obtains a "quieter" current as shown in Fig. 6.

In further runs to be discussed, the plasma density varies from $10^4/cm^3$ to $10^5/cm^3$. The corresponding values of Debye length, Debye number, electron

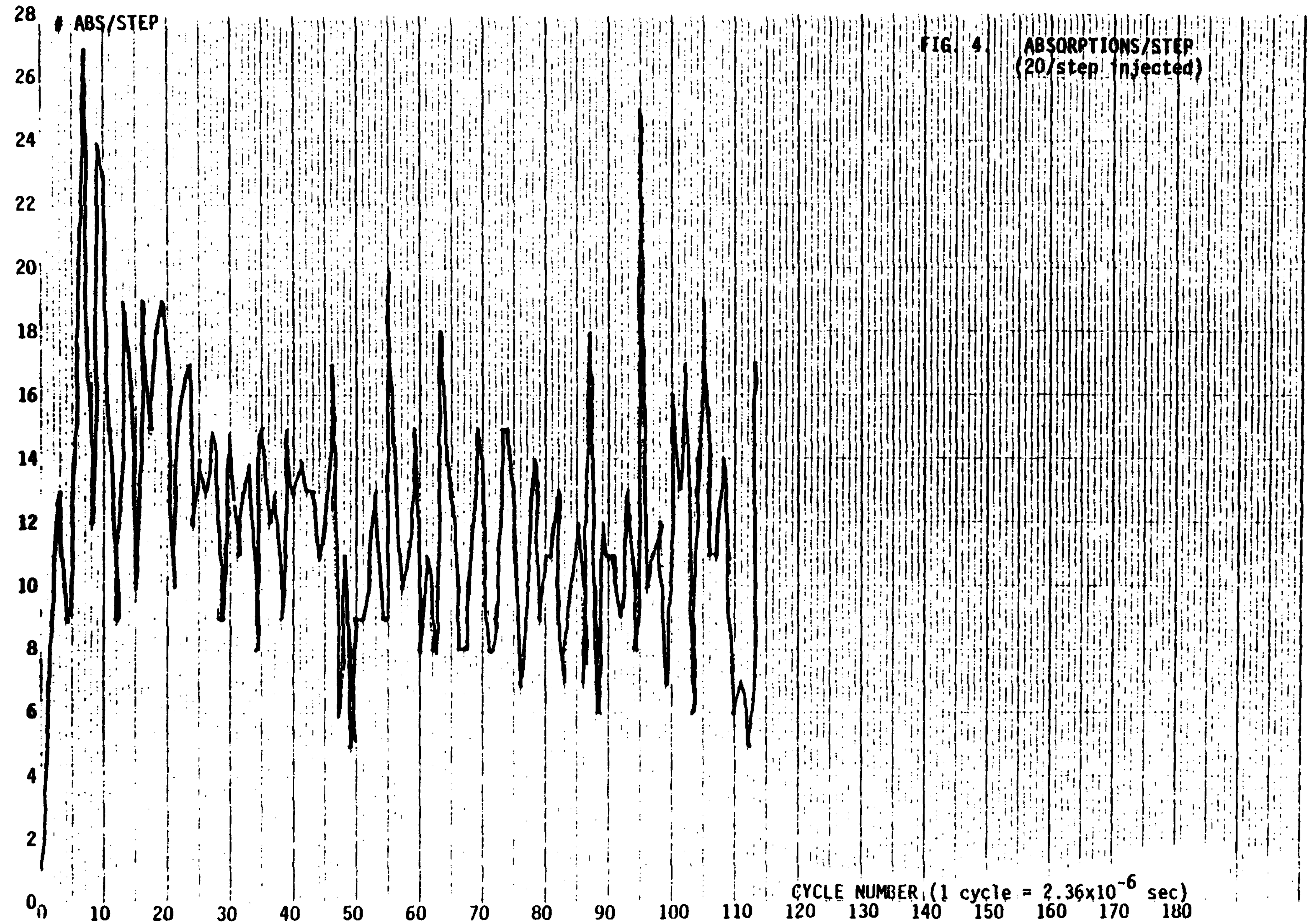


FIG. 4. ABSORPTIONS/STEP
(20/step injected)

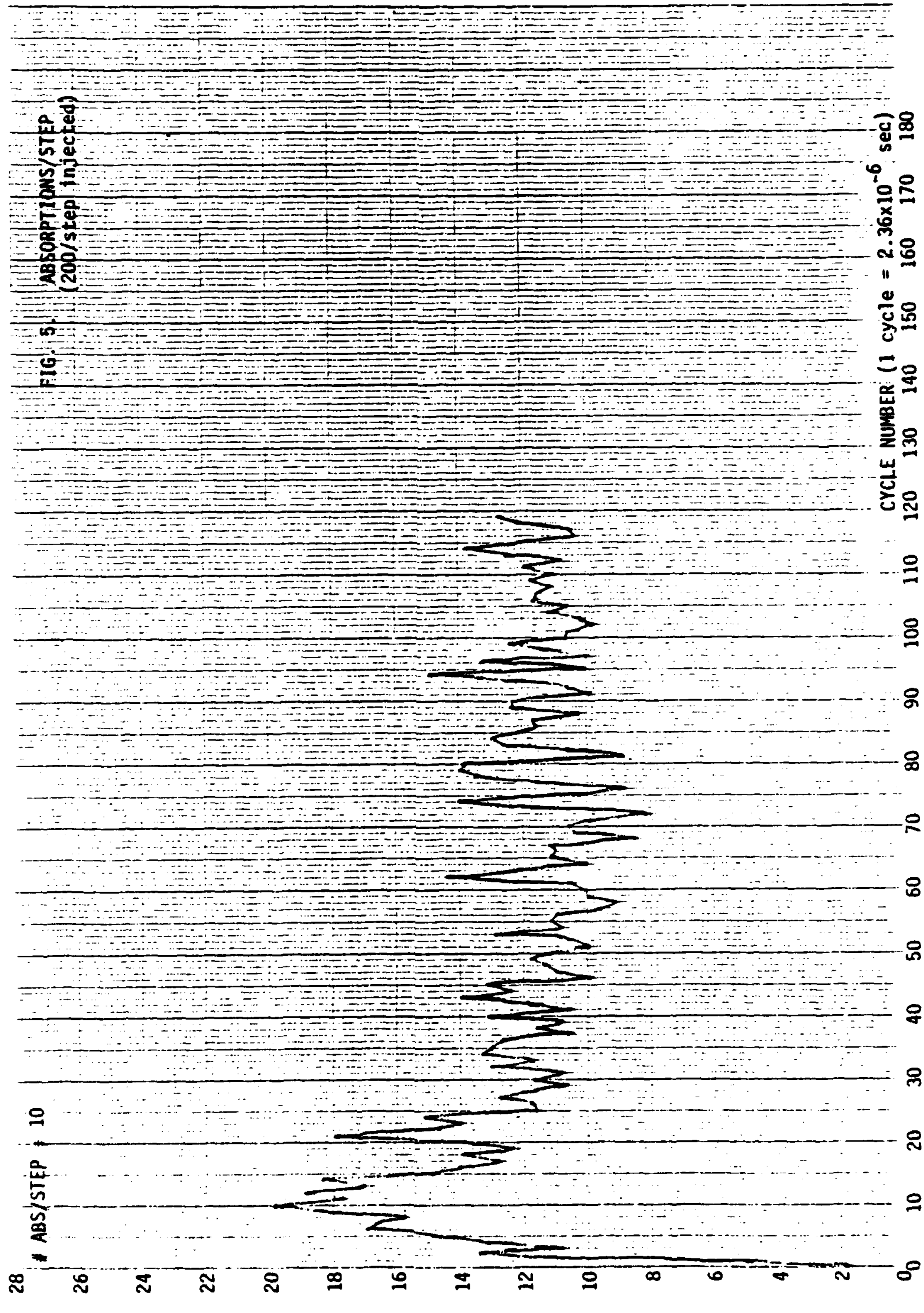


FIG. 5. ABSORPTIONS/STEP (200/step injected)

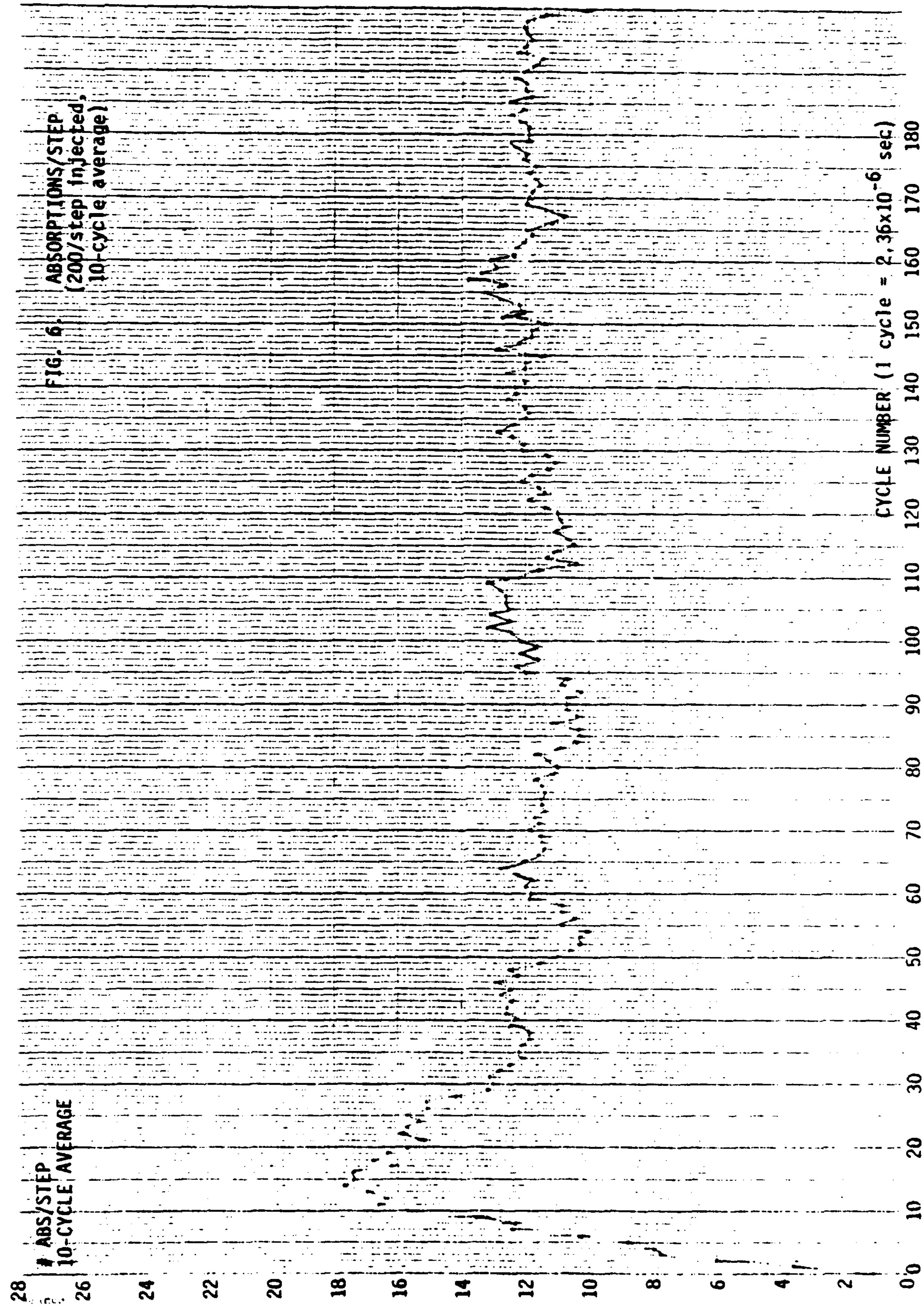


FIG. 6. ABSORPTIONS/STEP (200/step injected, 10-cycle average)

plasma period τ_e, and the ion plasma period τ_i, are given in the following table.

Run	n_0 (cm^{-3})	λ_D (cm)	$\lambda_D/50$ --	τ_e (s)	τ_i (s)
A	10^4	2.35	.047	1.11×10^{-6}	5.55×10^{-6}
B	2.5×10^4	1.49	.030	7.02×10^{-7}	3.51×10^{-6}
C	5.0×10^4	1.05	.021	4.96×10^{-7}	2.48×10^{-6}
D	7.5×10^4	0.86	.017	4.05×10^{-7}	2.03×10^{-6}
E	10^5	0.74	.015	3.51×10^{-7}	1.76×10^{-6}

In the following we will discuss Runs A and D in detail. Runs B and C gave results intermediate. Run E was so strongly fluctuating that it was not pursued further. The density of 7.5×10^4 seemed to be the largest (at T=0.1 ev) that could be treated at the present boundary condition (20 cm from the spacecraft surface). Higher densities will require that the boundary be moved inward.

Figure 7 (Figs. 7a-7g) shows results of Run A ($n_0=10^4$). This is a printer plot of the variations with cycle number (one cycle = 9.5×10^{-8} sec) of

A = ion absorptions per step (scaled from 0 to 30)

B = ion population (scaled from 2500 to 5000)

C = electron population (scaled from 2500 to 5000)

D = maximum potential (scaled from 0 volts to 25 volts)

The significances of A, B, C, D, are as follows. We inject 100 electrons and 20 ions per step.

A represents, through the ratio of the number absorbed to the number injected, multiplied by the ratio of the area of collection to the area of injection, the collected current relative to the ambient thermal current available.

B and C represent the ion and electron populations (numbers of particles active throughout the grid). These should become constant as the steady state is approached. In order to arrive at the steady state as quickly as possible, a "quiet start" approach is used, wherein the space is previously

PRINTEP PLOT SUMMARY OF DATA AS A FUNCTION OF TIME. DENCC = 1.0000E+04

FIG. 7a. RUN A

COARSE NEAREST GRID POINT MODEL

(Density $10^4/cm^3$)

SYMBOL CORRESPONDENCE

A	=	ION ABSORPTIONS	SCALED FROM	0.0	TO	30.0
B	=	ION POPULATION	SCALED FROM	2500.0	TO	5000.0
C	=	ELECTRON POPULATION	SCALED FROM	2500.0	TO	5000.0
D	=	MAXIMUM POTENTIAL	SCALED FROM	0.0	TO	25.0

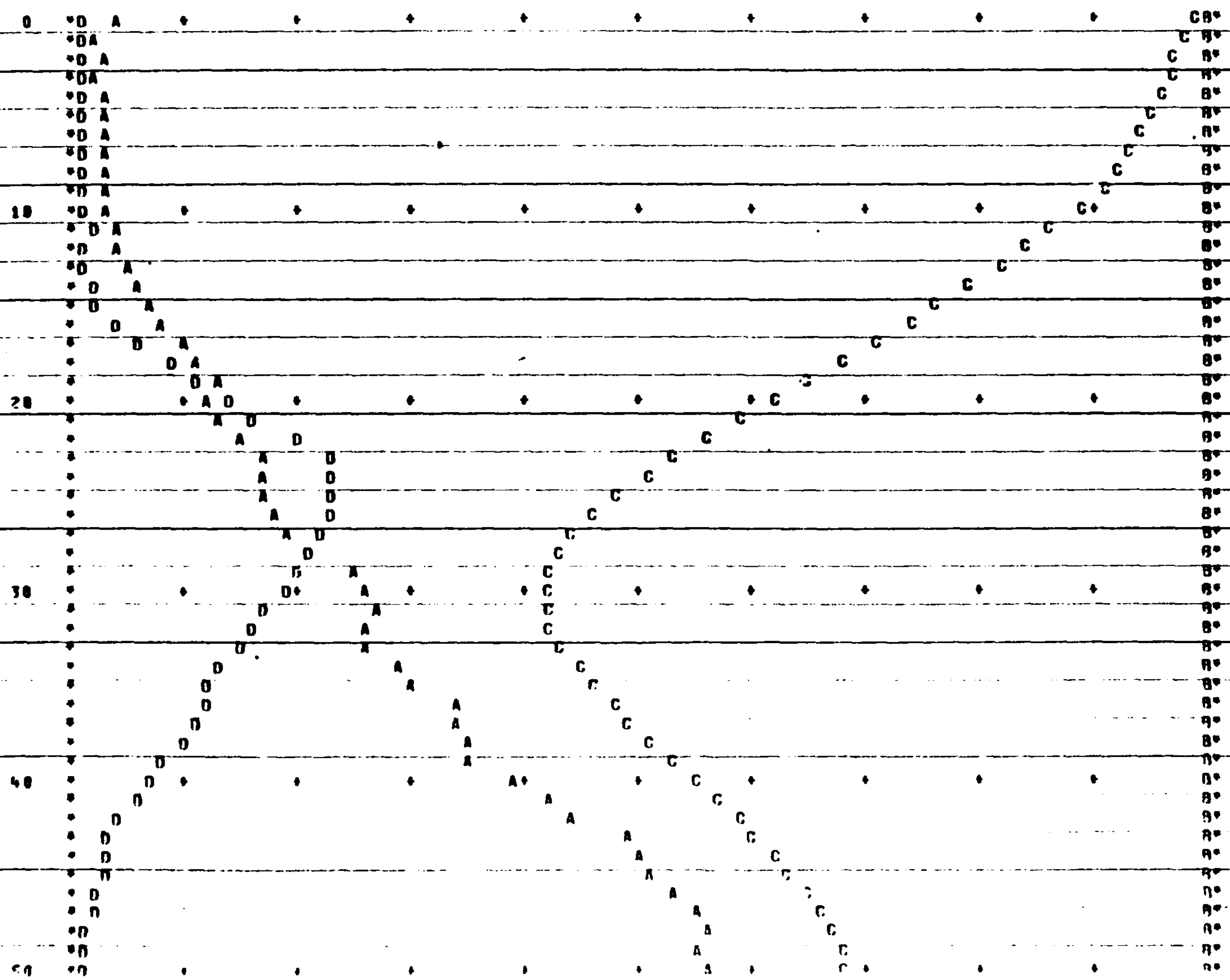


FIG. 7a.

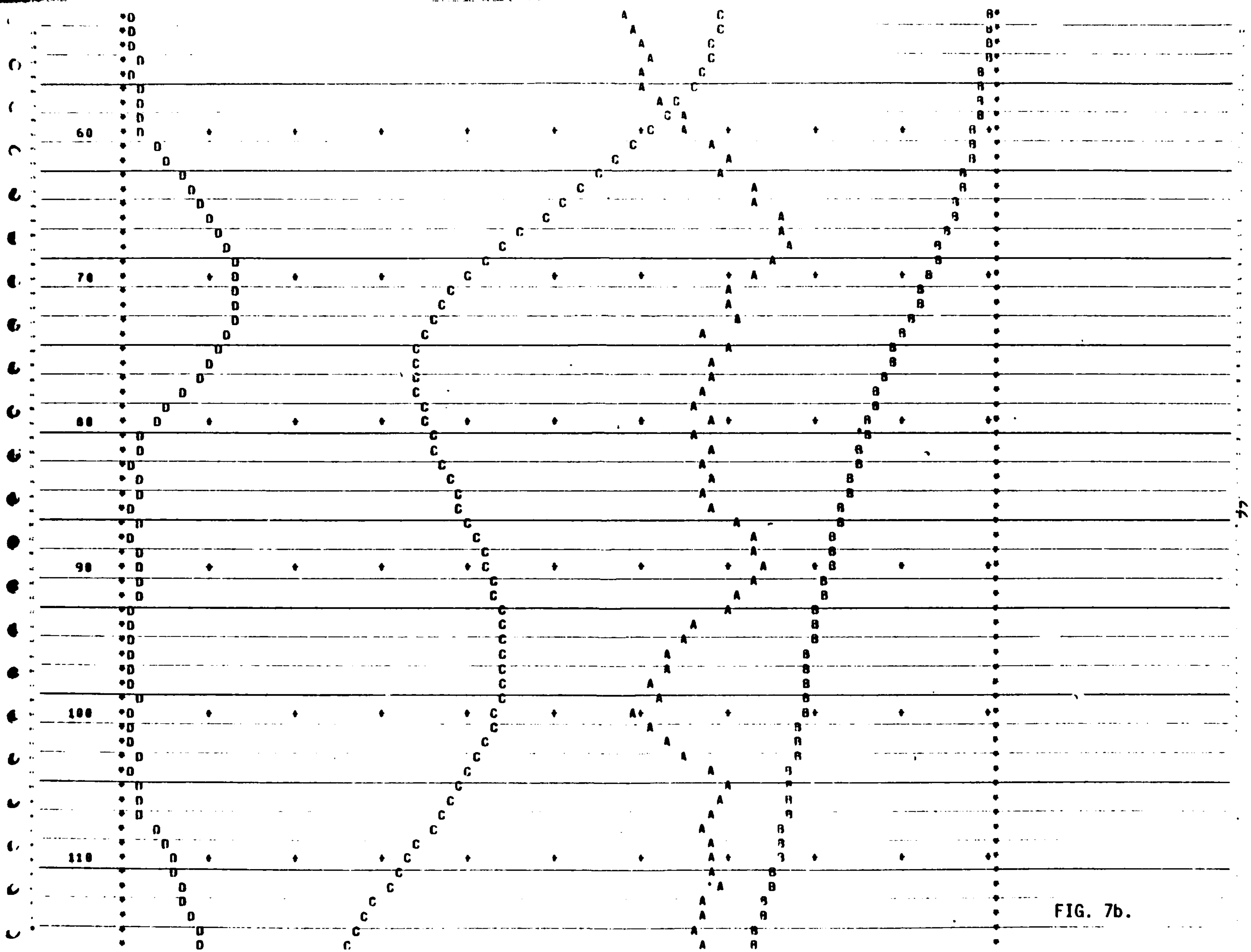


FIG. 7b.

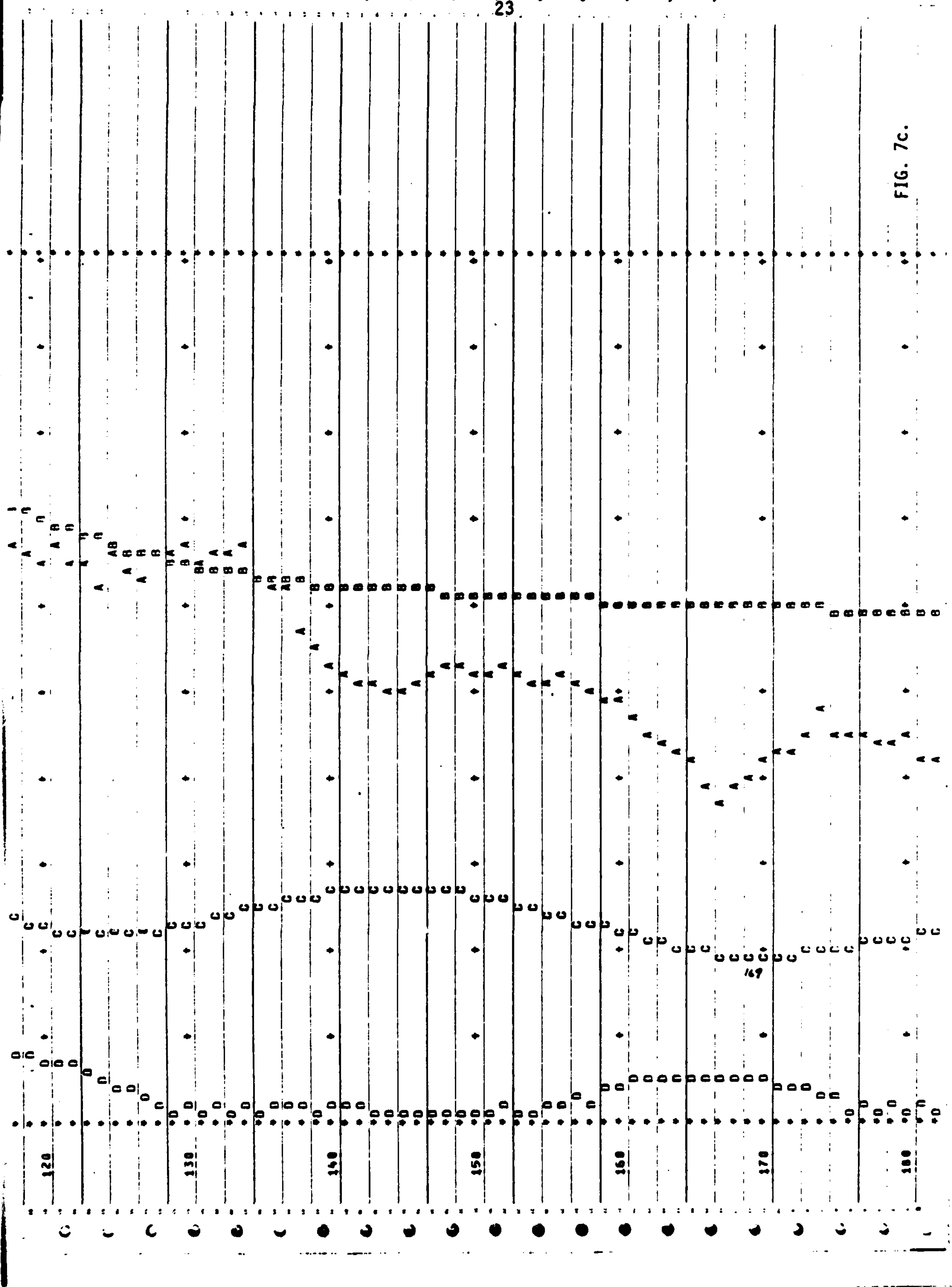


FIG. 7c.

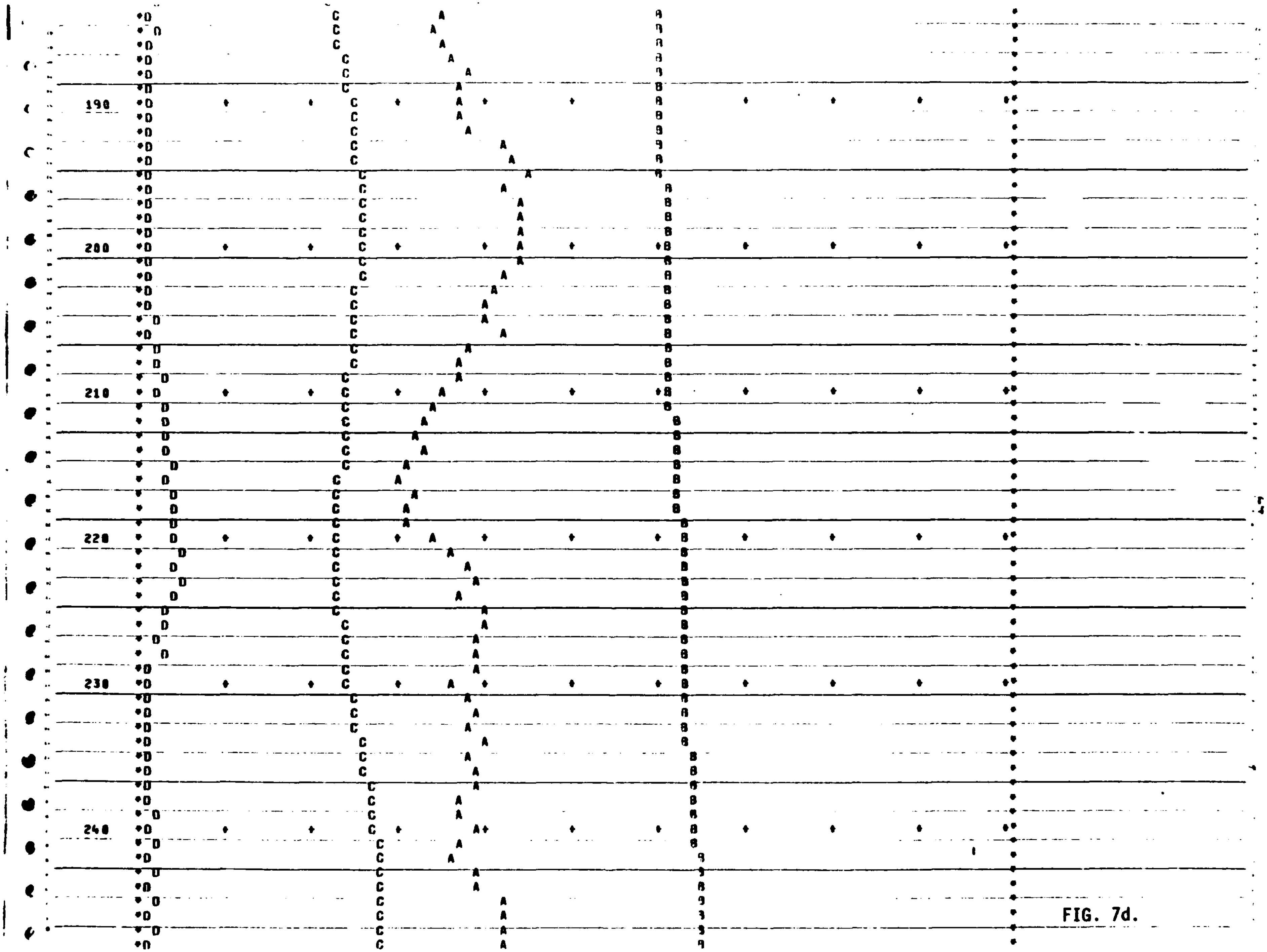


FIG. 7d.

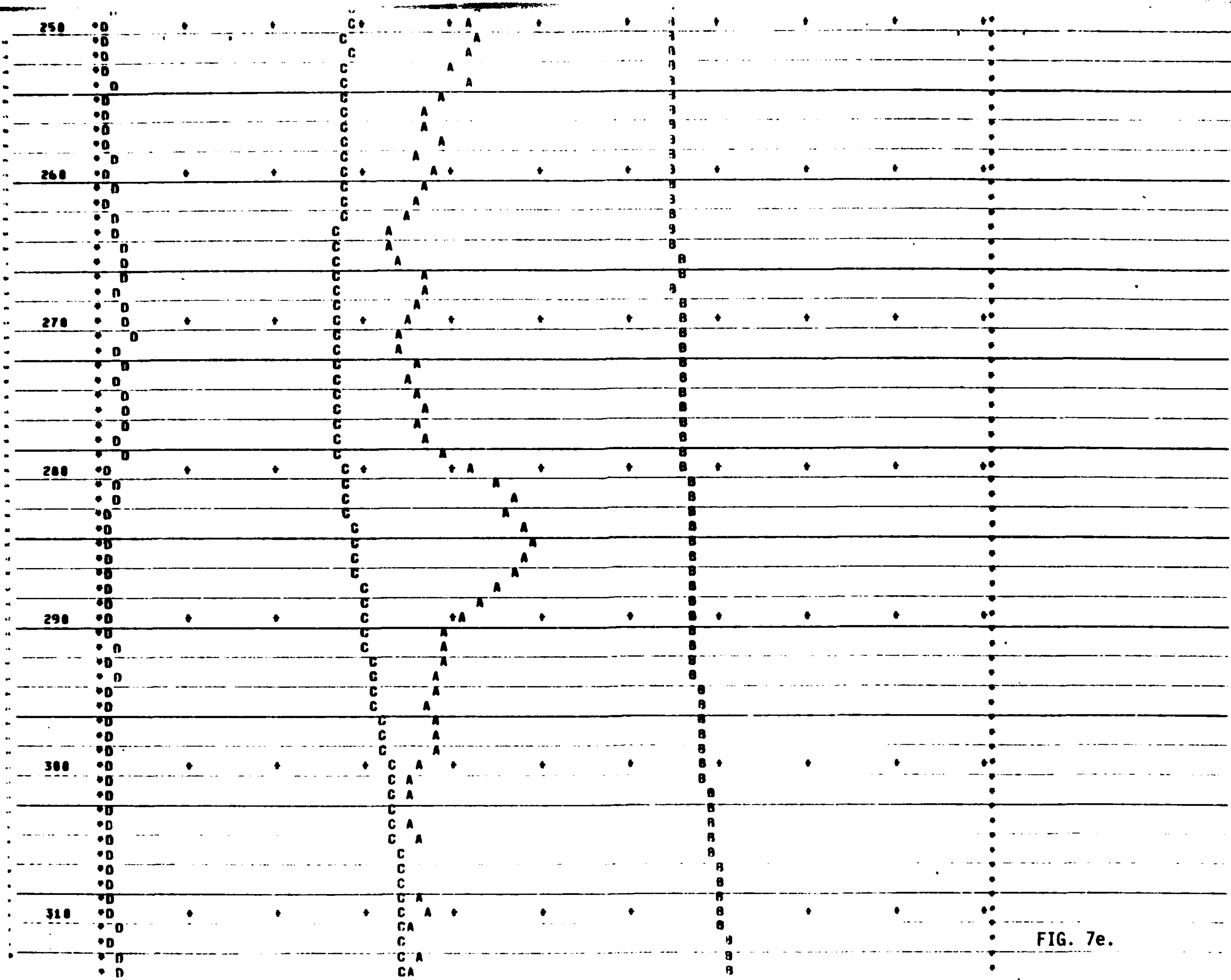


FIG. 7e.

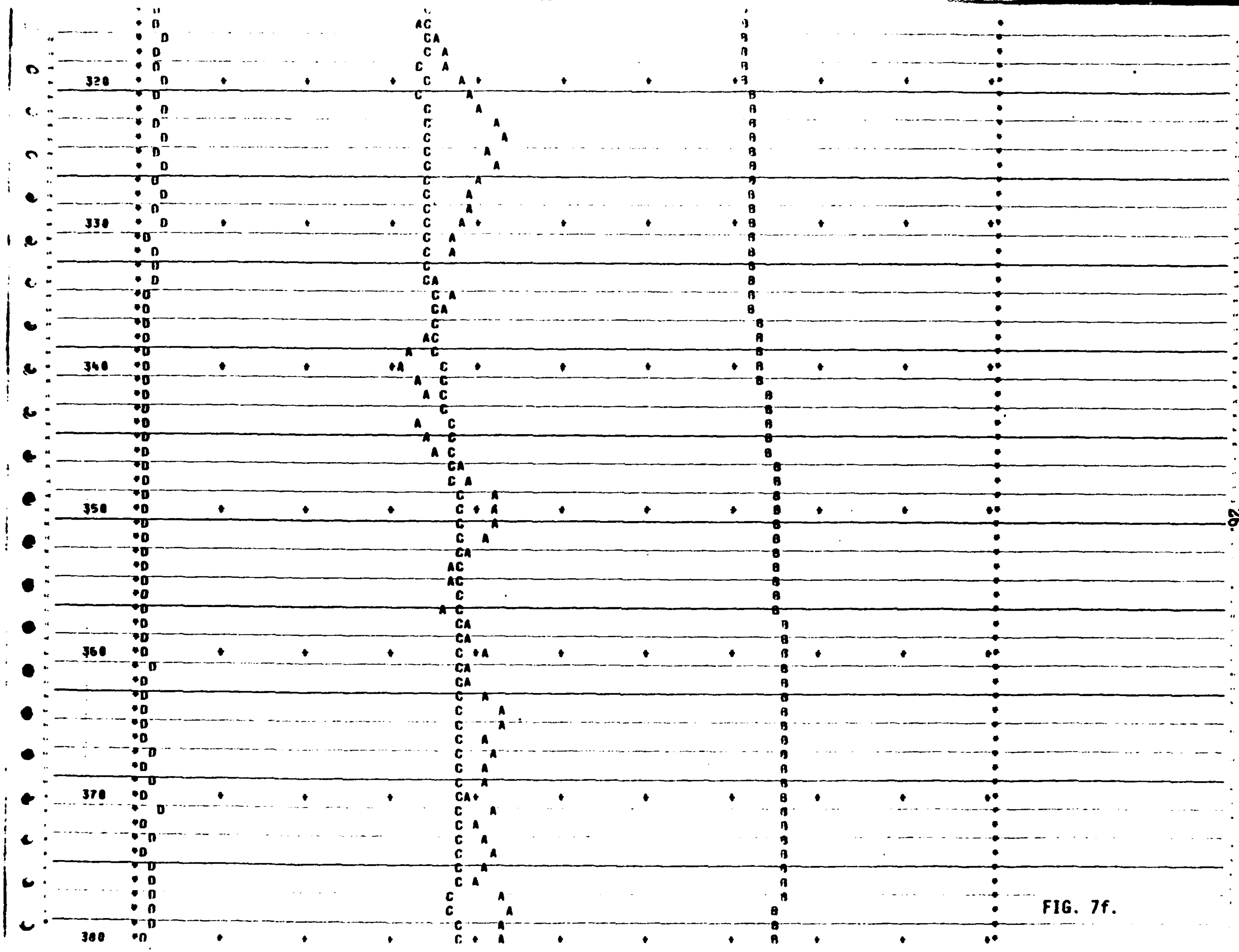


FIG. 7f.

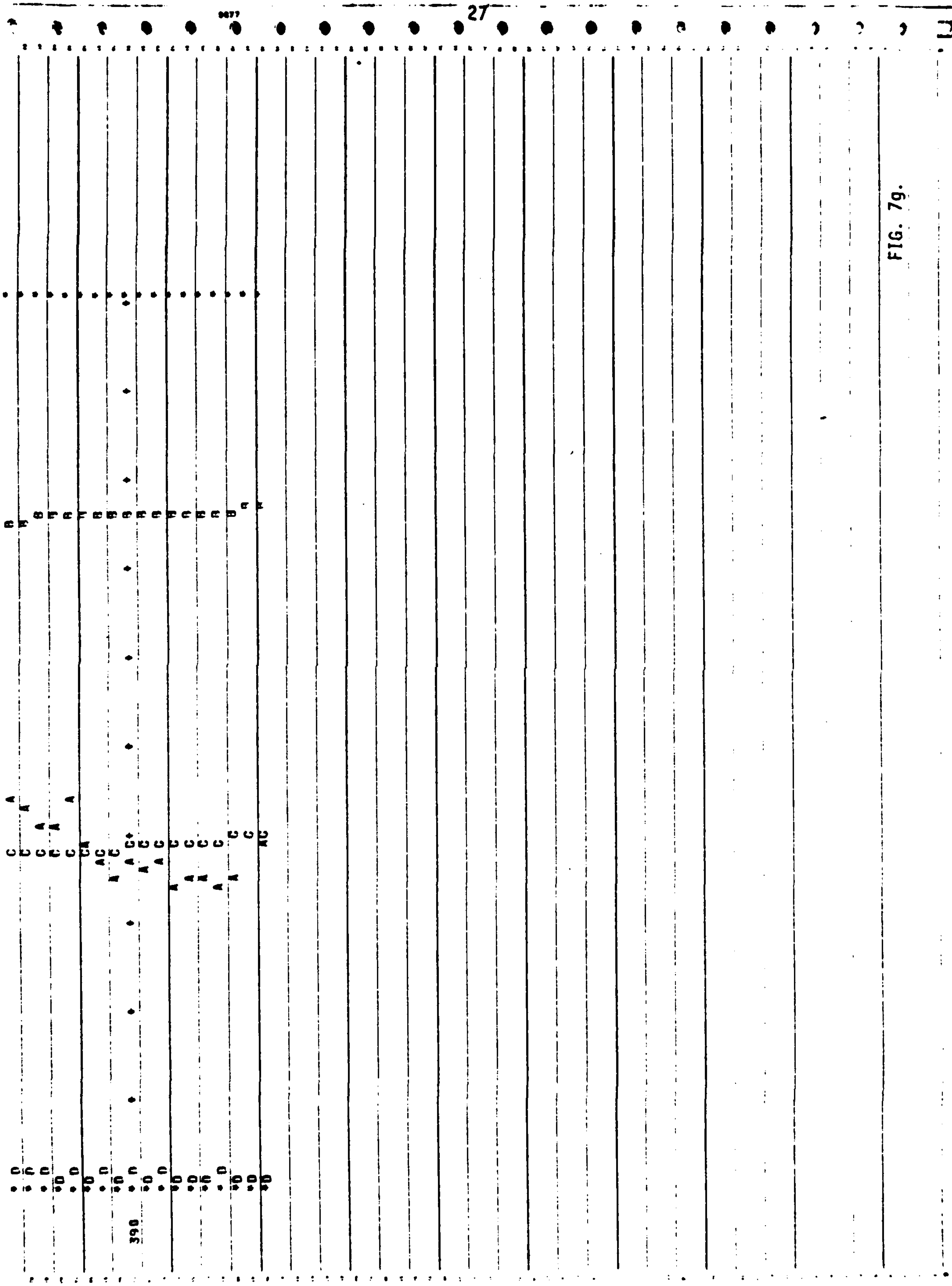


FIG. 7g.

filled with particles so as to represent normal density. In the present case 5000 ions and 5000 electrons are used, to represent the unperturbed populations.

D is a measure of the fluctuation intensity. Since the boundary potentials are allowed to float, they are susceptible to oscillations. In particular, when the instability is severe the boundary potentials tend to become positive, with the maximum increasing as the severity increases. Hence the maximum is monitored.

The behavior of A, B, C, and D as functions of time are shown by Fig. 7 as follows. Large transients occur in all 4 quantities during the first 150 cycles ($t=1.4\times10^{-5}$ sec). Both the electron (C) and ion (B) populations decrease from their unperturbed values, the electrons because they are repelled away, and the ions because they are accelerated and spend less time in any given volume (a consequence of Langmuir probe theory). The electrons (C) oscillate while approaching their final value, the oscillations having diminishing amplitude. The period of the oscillations is roughly 50 cycles, or 4.7×10^{-6} sec, about 4 times the electron plasma period (see table), approximately equal to the ion plasma period. It is not clear whether the oscillations represent physical behavior; their smoothness suggests that they do. However, the effects of numerical "aliasing", wherein high-frequency sources can drive low-frequency oscillations through the finite grid spacing, must certainly be present because of the relatively large grid intervals (of order 10 cm).

After about 220 cycles (2.1×10^{-5} sec) the electron population has settled down and subsequently rises slowly toward its asymptotic value, about 3500 out of the original 5000. The ions (B), on the other hand, do not oscillate, but drop monotonically toward a shallow minimum at about 180 cycles (1.7×10^{-5} sec), and then rise slowly toward their asymptotic population, about 4400 out of the original 5000.

The ion absorption rate (A) rises initially, overshooting and subsequently decaying to an asymptotic value of about 11 absorbed per step. A higher-frequency oscillation of relatively small amplitude and a period of about 20 cycles (1.9×10^{-6} sec) is superimposed. The ratio of the current collected to

the unperturbed current that would be collected in the absence of electric fields is given by dividing the number absorbed per step by the number injected per step, multiplied by the ratio of injection area to collection area, and divided by DELTA, a time-step input defined earlier.

Figure 8 (Figs. 8a-8g) shows results of Run D (n_0=7.5x10^4). All inputs are the same as in Fig. 7. However, now we see that there are severe fluctuations in curve D, the boundary potential. The ion and electron populations (B and C) both descend almost monotonically to lower values than they had in Fig. 7. They are closer to each other (about 3300 and 3000) than they were. The ion absorption rate (A) initially overshoots to a large value, but more quickly settles to a "steady" state, with some fluctuations though not severe, about 6 absorbed per step, roughly half of that in Fig. 7.

During the transient period in absorptions (first 70 cycles or about 6.6x10^{-6} sec), the electron population (C) oscillates with relatively small amplitude and smaller period (about 25 cycles or 2.4x10^{-6} sec) than in Fig. 7. This period is again about the same as the ion plasma period (table). Hence this oscillation, which appears regular, may be physical. It is interesting that the electron population fluctuations (the ions have little or none) are much less severe than and apparently uncorrelated with the boundary potential fluctuations (D).

It should be stated that a "cloud-in-cell" model was also developed and used, without significantly reducing the noise as had been hoped. It also gave slightly different (higher) values for the steady state absorption rate. The runs made with the cloud model will not be discussed.

Figure 9 shows a potential contour plot for Run D, generated by the computer printer, at the "steady" state.

PRINTER PLOT SUMMARY OF DATA AS A FUNCTION OF TIME. DENCC = 7.5000E+04
COARSE NGP MODEL

FIG. 8a. RUN D

(Density $7.5\times10^4/cm^3$)

SYMBOL CORRESPONDENCE

A =	ION ABSORBSIONS	SCALED FROM	0.0	TO	30.0
B =	ION POPULATION	SCALED FROM	2500.0	TO	5000.0
C =	ELECTRON POPULATION	SCALED FROM	2500.0	TO	5000.0
D =	MAXIMUM POTENTIAL	SCALED FROM	0.0	TO	25.0

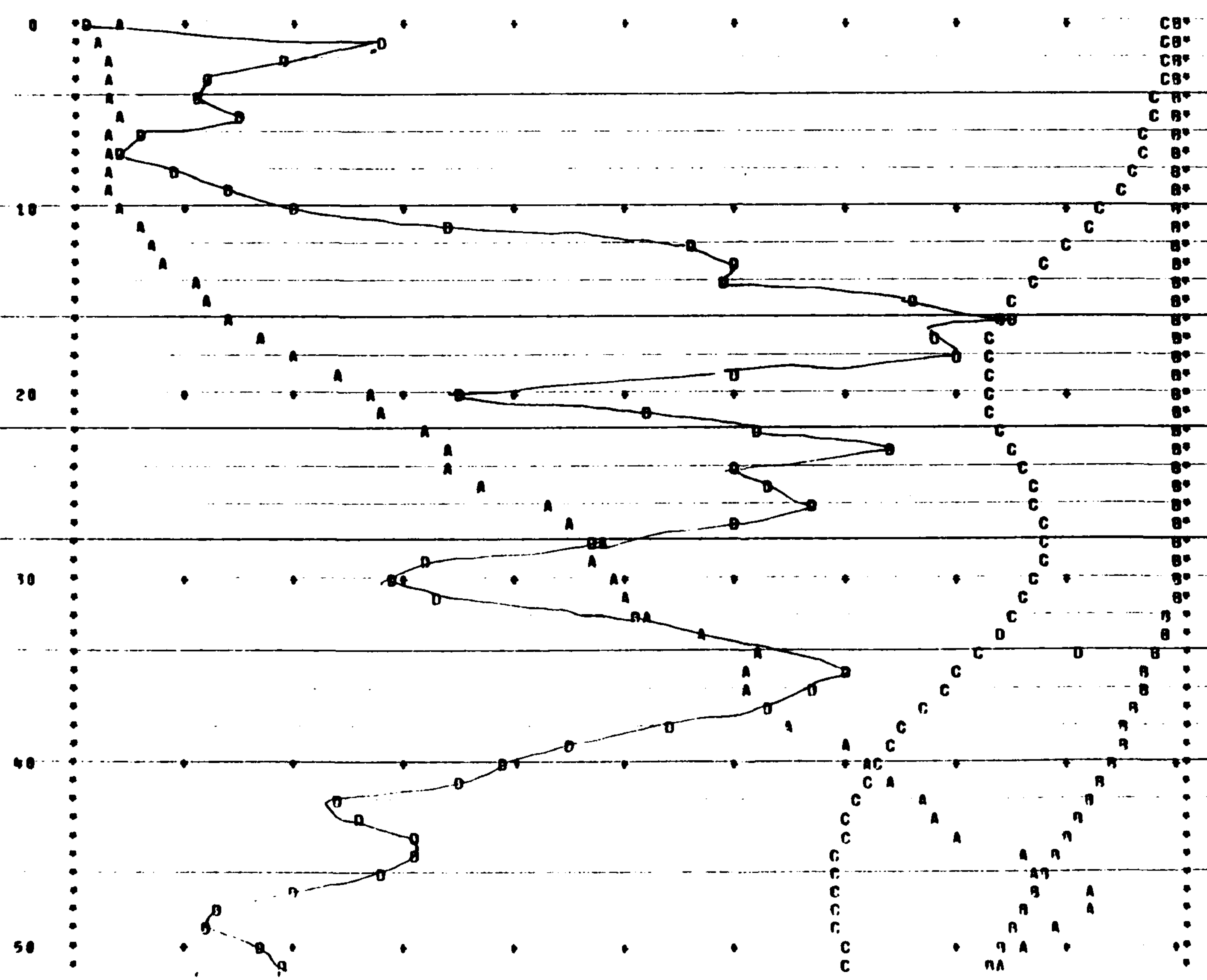


FIG. 8a.

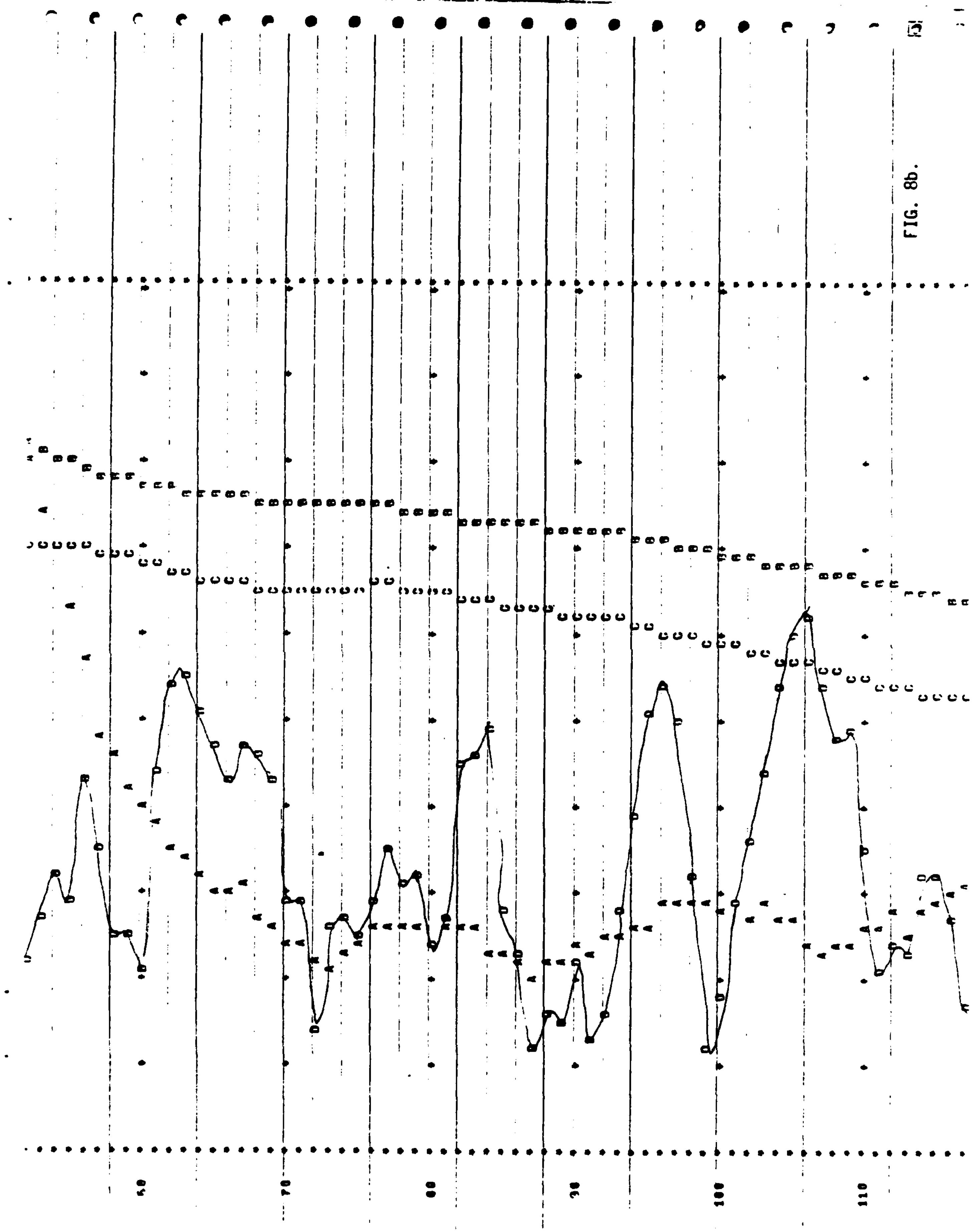


FIG. 8b.

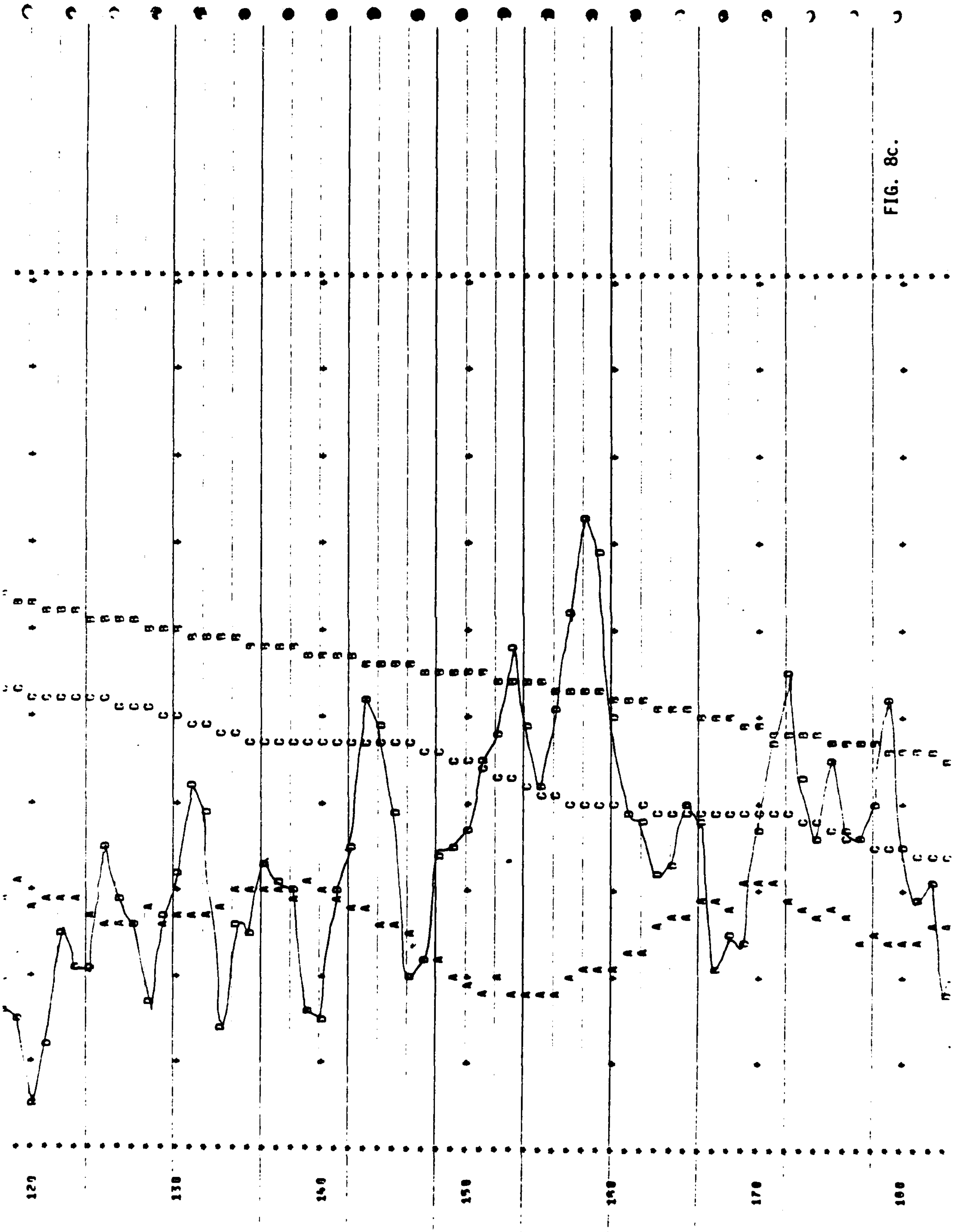


FIG. 8c.

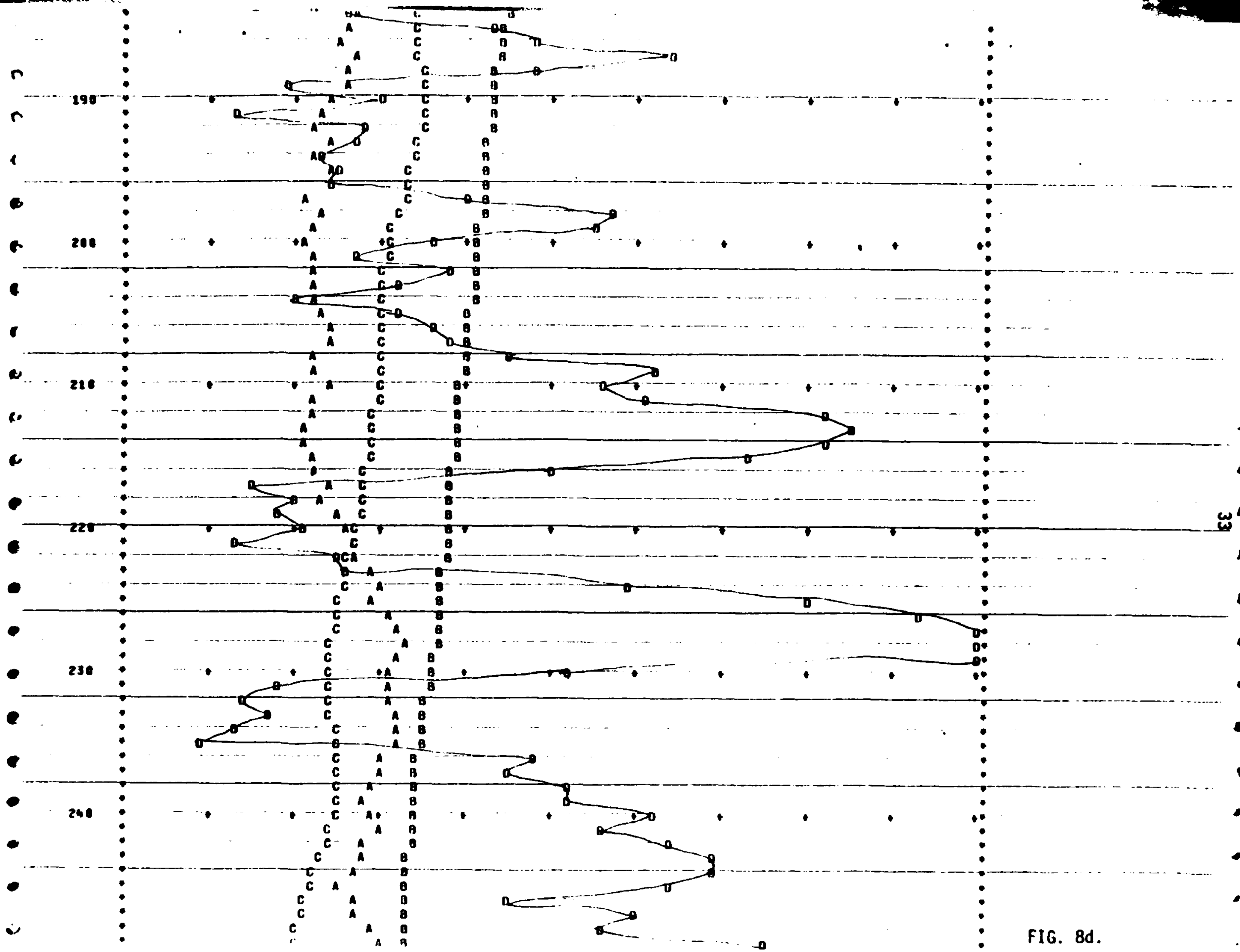


FIG. 8d.

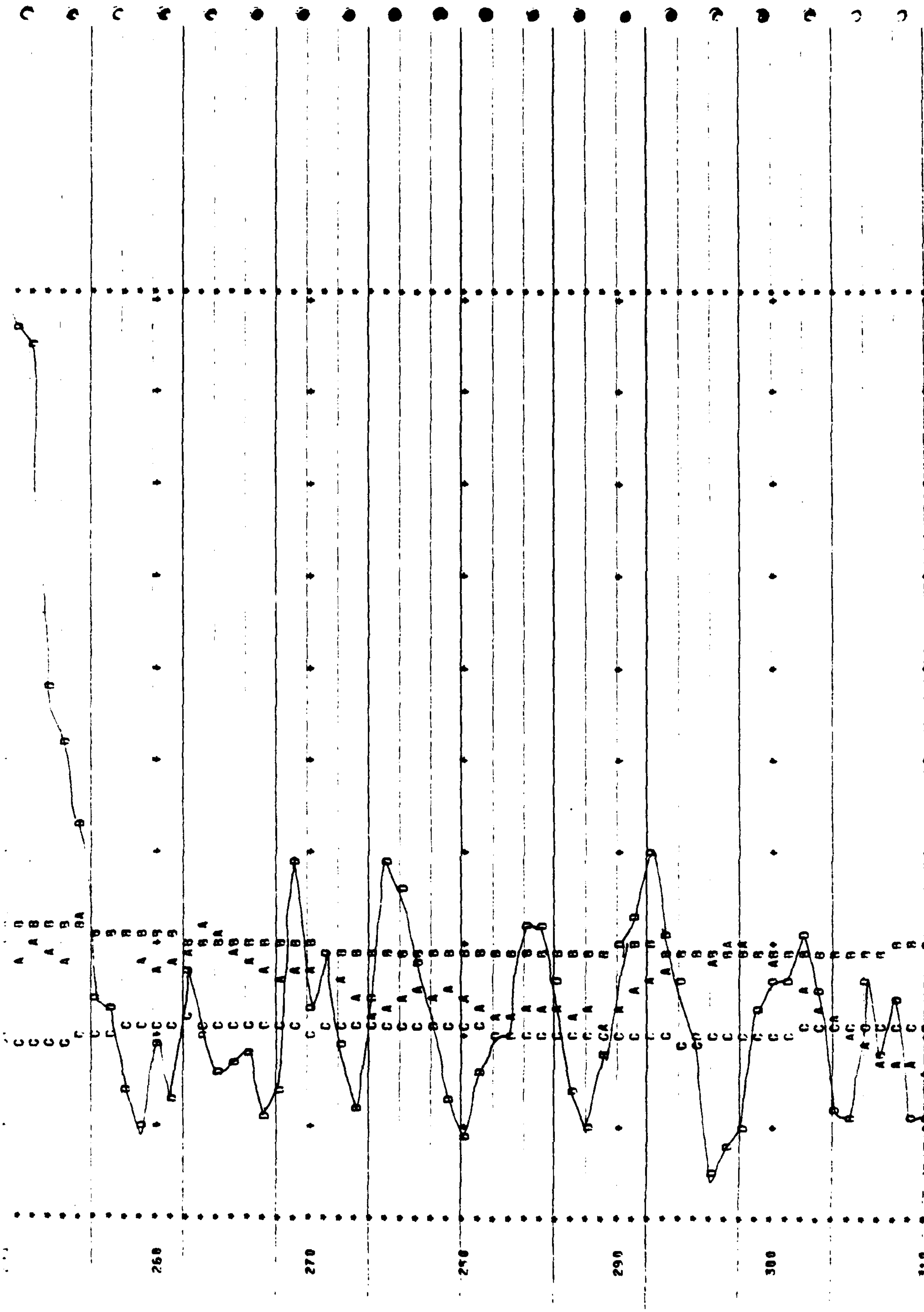


FIG. 8e.

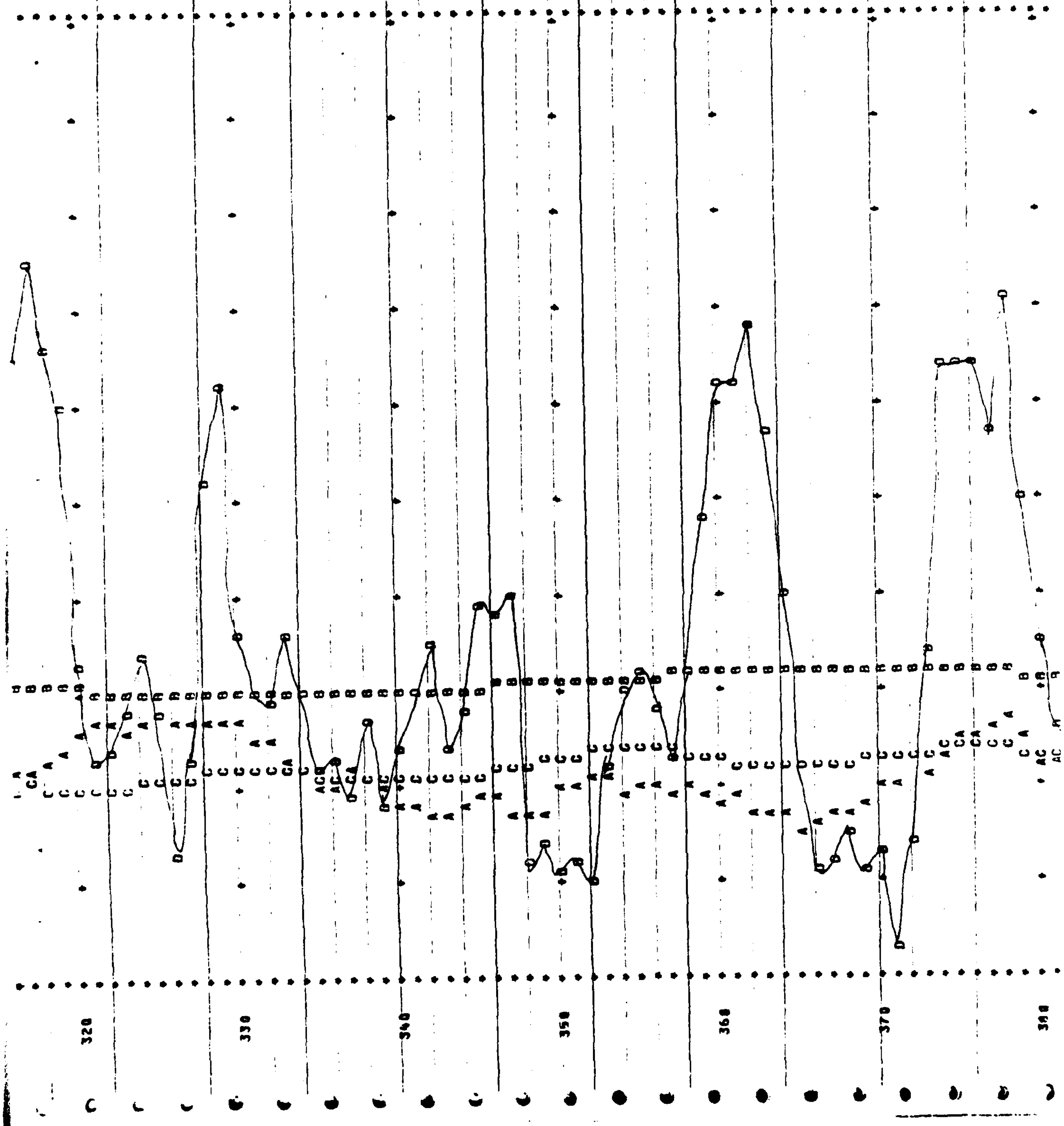


FIG. 8f.

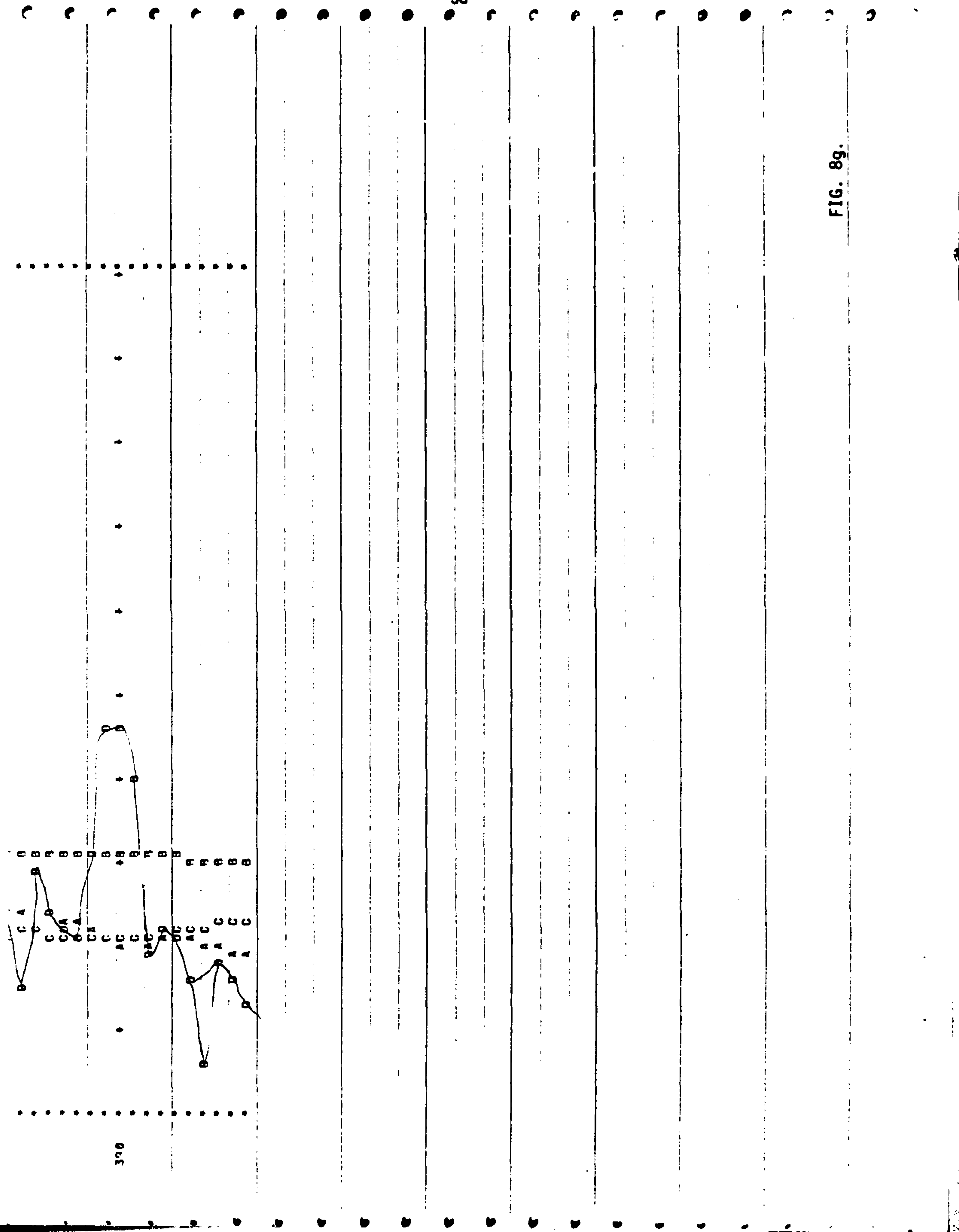


FIG. 8g.

FIG. 9.

EQUIPOTENTIAL CONTOURS

J = 0 to +1

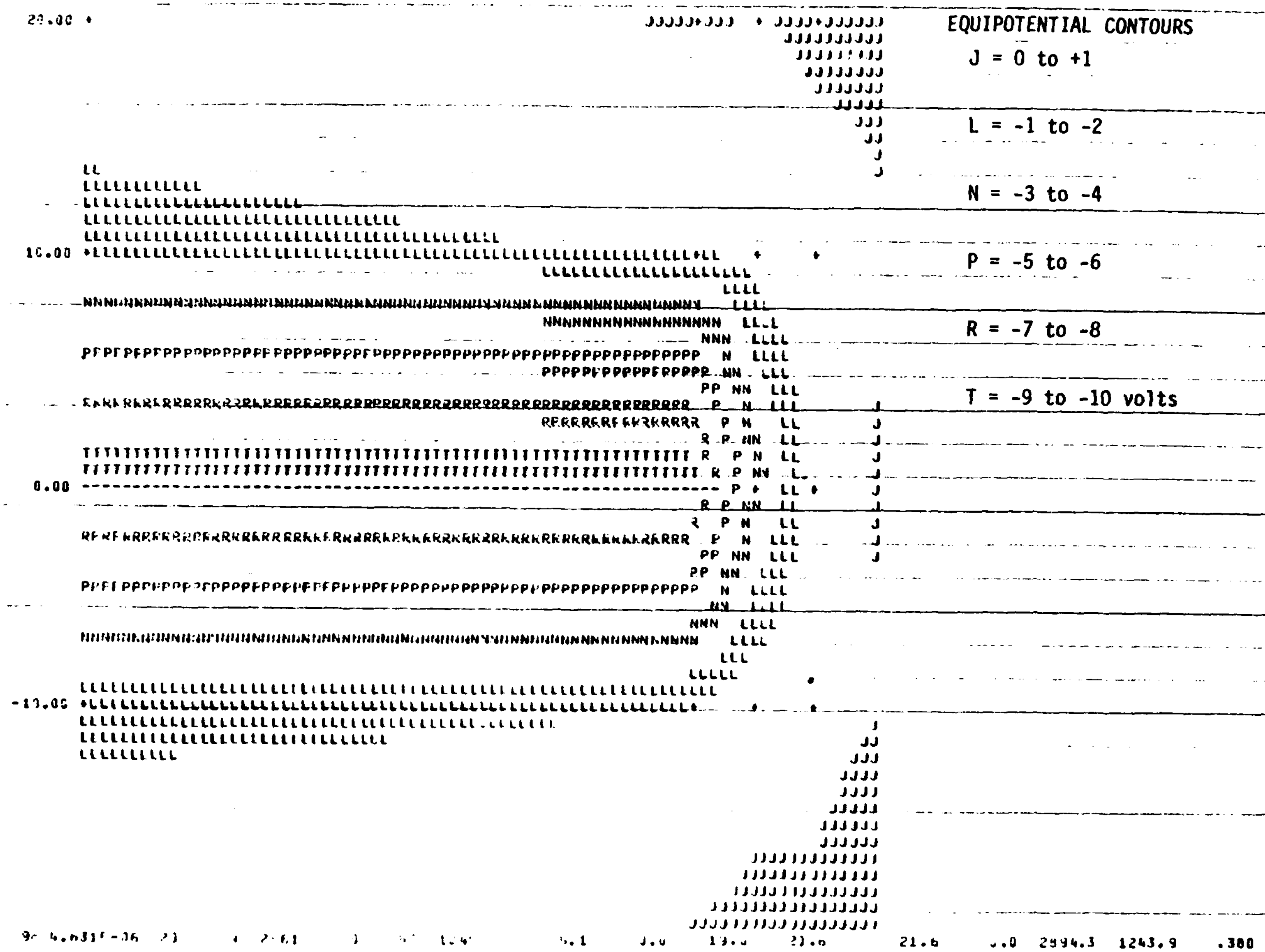
L = -1 to -2

N = -3 to -4

P = -5 to -6

R = -7 to -8

T = -9 to -10 volts



PARKTDC PROGRAM

The listing for the PARKTDC computer program is given in the following pages. (PARKTDC = Parker Time Dependent Charging.)

LIBRARY INDEX 02/11/80

```
      PROGRAM PARKTDC(INPUT=65,OUTPUT=513,TAPE1=513,TAPE2=513,
     1 TAPE3=513,TAPE5=INPUT,TAPE6=OUTPUT)
C***********************************************************************
C
C     TIME-DEPENDENT CHARGING AND SPACECRAFT SHEATH IN
C     R-Z GEOMETRY
C
C     LEE W. PARKER /// ERNEST G. HOLEMAN.     LEE W. PARKER, INC.
C         252 LEXINGTON ROAD, CONCORD MA. 01742
C*****
C  VERSION -- E --.   MODE SELECTIBLE MODEL.
C
C     MODE = 0 ==>  STANDARD CALCULATION MODEL
C     MODE = 1 ==>  CLOUD MODEL
C     MODE = 2 ==>  WEIGHTED PARTICAL MODEL
C
C    E. HOLEMAN.  FROGRAMMER.  LAST ALTERED  2 MAR 79
C*****
C***********************************************************************
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 MONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51) ,
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      COMMON/DEN/RPT,ZPT,AL1,BE1,EV,TVOLTS,DENST,NPTS,SPEED0,CURR,
     1 IPART,PARTCL(2),DELTA,SPEED,QDISK,DZMIN,DTSEC,NINJ,
     2 VSAVE(10,100),TVCHG,NRAN,NCESC,NCABS,NCACT,NCHGE(11),NCHGI(11),
     3 KOUNT(51),XKOUN(51),MODE,NCLOUD
      COMMON/FLD/DEBYE,DIELEC,IRELAX,PHI(11),DENCC
      DIMENSION DATE(20)
      DIMENSION ZA(11),ZB(11),ZAI(11),ZBI(11)
      DIMENSION PART1(2),PART2(2),X1(11),NETCH(51)
      DIMENSION KHISTI(51,12),KHISTE(51,12),AVGI(51),RMSDI(51),
     1 AVGE(51),RMSDE(51),KHABSI(12),KHACTI(12),
     2 KHABSE(12),KHACTE(12),LINE(14)
      DATA PART1/5H IONS, 5H     /,PART2/5H ELEC,5HTRONS/
      NF(IZ,JR)=JR +  NR*(IZ-1)
      L=5
      M=6
      IV=1
      JV=2
      KV=3
      PI=3.1415926535898
      EV=0.
      ELK=1H
      NRAN=0
      INIT=0
      NR0=80
      NZ0=55
C
C*****
C  22 JUN 78 TEMPORARIES FOR INTERACTIVE MODE
      CALL SSWTCH(6,ISW6)
```

```
      IF(ISW6.NE.1) GO TO 1000
      CALL DISCON(M)
      CALL DISCON(L)
      REWIND M
      REWIND L
      REWIND KV
C*****
C
1000  READ(L,9999) DATE
9999  FORMAT(20A4)
      IF(EOF(L)) 99,1001
1001  WRITE(M,9998) DATE
9998  FORMAT(1H1,26H TIME DEPENDENT CHARGING. ,20A4)
C
C     GEOMETRIC PARAMETERS
C
      READ(L,1111) JEDGE,NR,NZA,NZB,IRELAX,MODE
      NZ=NZA + NZB - 1
      NTOT=NZ*NR
      READ(L,2222) (RR(J),J=1,NR)
      READ(L,2222) (ZA(I),I=1,NZA)
      READ(L,2222) (ZB(I),I=1,NZB)
C
      READ(L,2222) (PHI(J),J=1,JEDGE)
      JEDGEP=JEDGE+1
      RADIUS=.5*(RR(JEDGEP) + RR(JEDGE))
      WRITE(M,9990) JEDGE,NR,NZA,NZB,IRELAX
      IF(MODE.EQ.0) WRITE(M,9991) MODE
      IF(MODE.EQ.1) WRITE(M,9992) MODE
      IF(MODE.EQ.2) WRITE(M,9993) MODE
      WRITE(M,9980) RADIUS,(J,RR(J),J=1,JEDGE)
      WRITE(M,9970) (J,RR(J),  J=JEDGEP,NR)
      WRITE(M,9960) (I,ZA(I),I=1,NZA)
      WRITE(M,9950) (I,ZB(I),I=1,NZB)
      WRITE(M,9940) (J,PHI(J),J=1,JEDGE)
C
C     ITERATION AND SPACE CHARGE OPTIONS
C
      IT=0
      READ(L,1111) NPRINT,NPTS,IT,ITS,NIINJ,ISKIP,JSKIP,MASS
      WRITE(M,9930)NPRINT,NPTS,IT,ITS,NIINJ,ISKIP,JSKIP,MASS
      IF(IT.GT.0) READ(L,3333) NRAN
      READ(L,2222) DELTA,TIME,TVIONS,TVELEC,DENCC,WCLOUD
      IF(IT.GT.0) READ(L,1111) (NCHGI(I),I=1,JEDGE)
      IF(IT.GT.0) READ(L,1111) (NCHGE(I),I=1,JEDGE)
      IF(IT.GT.0) READ(L,2222) (Q(I),I=1,NTOT)
      ITMAX=IT+ITS
      DIELEC=1.
C     DEBYE=0.
C     DEBYE= 743.39*SQRT(TVELEC/DENCC)/RADIUS
C
```

```
      IF(NPTS.EQ.0) READ(L,2222) RPT,ZPT,AL1,BE1,EV
      IF(NPTS.EQ.0) WRITE(M,9880) RPT,ZPT,AL1,BE1,EV
      IF(NPTS.EQ.0) TIME=0.
      DO 200 I=1,NZA
200   ZZ(I)=ZA(I)
      DO 210 I=2,NZB
      I1=NZA+I-1
210   ZZ(I1)=ZB(I)
C
      WRITE(M,9870) (I,ZZ(I),I=1,NZ)
C
C  CALCULATE INTERSTITIAL GEOMETRY
C
      IR=NR+1
      IZA=NZA+1
      IZB=NZB+1
      IZ=NZA+NZB
      IZZ=1
C
      RI(1)=RR(1)
      RI(IR)=RR(NR)
      DO 300 J=2,NR
300   RI(J)=.5*(RR(J-1)+RR(J))
C
      ZAI(1)=ZA(1)
      ZAI(IZA)=ZA(NZA)
      ZCI(1)=ZZ(1)
      ZCI(IZ)=ZZ(NZ)
      DO 350 I=2,NZA
      ZAI(I)=.5*(ZA(I-1)+ZA(I))
      IZZ=IZZ+1
      ZCI(IZZ)=ZAI(I)
  350 CONTINUE
C
      ZBI(1)=ZB(1)
      ZBI(IZB)=ZB(NZB)
      DO 400 I=2,NZB
      ZBI(I)=.5*(ZB(I-1)+ZB(I))
      IZZ=IZZ+1
      ZCI(IZZ)=ZBI(I)
  400 CONTINUE
C
      WRITE(M,9860) (J,RI(J),J=1,IR)
      WRITE(M,9850) (I,ZAI(I),I=1,IZA)
      WRITE(M,9840) (I,ZBI(I),I=1,IZB)
      WRITE(M,9835) (I,ZCI(I),I=1,IZ)
C
      DO 450 N=1,NTOT
      KOUNT(N)=0
      XKOUN(N)=0
      IF(IT.EQ.0) X(N)=0.
```

```
      IF(IT.EQ.0) Q(N)=0.
  450 CONTINUE
C
      DO 500 J=1,NR
      R1=RI(J)
      IF(J.LT.NR) R2=RI(J+1)
      IF(J.EQ.NR) R2=RR(NR)
      AREAH(J)=PI*(R2**2 - R1**2)
      NCHGE(J)=0
      IF(J.GT.JEDGE) GO TO 500
      IF(IT.EQ.0.AND.IRELAX.GT.0) PHI(J)=0.
  500 CONTINUE
C
      DZMIN=10.**5
      DO 525 I=1,NZA
      Z2=ZAI(I)
      IF(I.LT.NZA) Z1=ZAI(I+1)
      IF(I.EQ.NZA) Z1=ZAI(IZA)
      DELZ(I)= Z2 - Z1
      DZMIN=AMIN1(DZMIN,DELZ(I))
  525 CONTINUE
C
      XMASS=MASS
      IZZ=NZA
      DO 550 I=1,NZB
      IZZ=IZZ+1
      Z2=ZBI(I)
      IF(I.LT.NZB) Z1=ZBI(I+1)
      IF(I.EQ.NZB) Z1=ZBI(IZB)
      DELZ(IZZ)= Z2 - Z1
      DZMIN=AMIN1(DZMIN,DELZ(IZZ))
  550 CONTINUE
C
      IFIRST=0
      JFIRST=0
C
C     MONMAX=0   FOR   MONOENERGETIC BEAM
C     MONMAX=1   FOR   ISOTROPIC MONOENERGETIC
C     MONMAX=2   FOR   ISOTROPIC MAXWELLIAN
C
      MONMAX=2
      DENOM=1.
      IF(MONMAX.EQ.1) DENOM=4.
      IF(MONMAX.EQ.2) DENOM=2.*SQRT(PI)
C
      PCON=PI*DENGG*DELTA*DZMIN*RR(NR)**2
      IF(MONMAX.GT.0) PCON=PCON*(1.+(ZZ(1)-ZZ(NZ))/RR(NR))*2.
      CHCON=1.6E-7*PCON
      XIINJ=NIINJ
      COMPAR=PCON/XIINJ
      SPEEDE=5.93E7*SQRT(TVELEC)
```

```
      SPEEDI=5.93E7*SQRT(TVIONS/XMASS)
      CURRI=1.6E-7*DENCC*SPEEDI/DENOM
      CURRE=1.6E-7*DENCC*SPEEDI/DENOM
      PLASPI=1.11E-4/SQRT(DENCC/XMASS)
      PLASPE=1.11E-4/SQRT(DENCC)
      DEBYE=743.39*SQRT(TVELEC/DENCC)
C
C     SPEED IS SCALE VELOCITY
C
      SPEED=SPEEDI
      DTSEC=DELTA*DZMIN/SPEED*DENOM
C
      WRITE(M,9920) DELTA,DTSEC,TIME,TVIONS,TVELEC,DENCC,PLASPI,PLASPE,
     1 DEBYE,PCON,COMPAR
      IF(MODE.EQ.1) WRITE(M,9921) WCLOUD
C
      NFPP=(NTOT/300)+1
      DO 600 IPAGE=1,NFPP
      WRITE(M,9830)
      CALL LIST(2,IPAGE)
  600 CONTINUE
C
C  INITIALIZE MATRICES FOR ANALYSIS OF VARIANCES
C
      DO 610 I=1,NTOT
      AVGI(I)=0.
      AVGE(I)=0.
      RMSDI(I)=0.
      RMSDE(I)=0.
      DO 610 J=1,12
      KHISTI(I,J)=0
      KHISTE(I,J)=0
  610 CONTINUE
      DO 620 I=1,12
      KHABSI(I)=0
      KHACTI(I)=0
      KHABSE(I)=0
      KHACTE(I)=0
  620 CONTINUE
C
      IF(IT.GT.0) GO TO 650
C
C     INITIALIZE SPACE CHARGE MATRIX
C       ASSUME QUIET START
C
      INIT=1
      VOL=PI*RR(NR)**2*(ZZ(1)-ZZ(NZ))
      RPART=DENCC*VOL
      NINJ=RPART/COMPAR
      CALL CONNEC(M)
      WRITE(M,*) "  INITIAL NINJ,RPART,COMPAR = ",NINJ,RPART,COMPAR
```

```
      CALL DISCON(M)
      DO 630 IDUM=1,2
      IPART=IDUM
      SPEED0=SPEEDI
      IF(IPART.EQ.2) SPEED0=SPEEDE
      CALL DENSTY(INIT)
      IFIRST=IFIRST+1
  630 CONTINUE
      INIT=0
C
  650 CONTINUE
C
C         COMPUTE IPRINT
C
      IPRINT=0
      IF(NPRINT.EQ.1) IPRINT=1
      IF(MOD(IT,JSKIP).LE.ISKIP-1) IPRINT=1
      IF(JFIRST.LE.9.OR.ITMAX-IT.LE.1) IPRINT=2
      IF(IT.EQ.0) IPRINT = 3
C
      IF(JFIRST.GT.0) TIME=TIME+DTSEC
      CALL FIELD
      IF(IPRINT.GT.1) WRITE(M,9790) IT
      IF(IT.GE.ITMAX.AND.ISW6.EQ.1) CALL CONNEC(M)
      CALL FPLOT(X,RR,ZZ,NR,NZ,NRC,NZO,RADIUS,XMAX,IT)
C
      IF(IT.GE.ITMAX) GO TO 1000
C
C
C     IPART LOOP - IPART=1,2, AND 3 FOR IONS, ELECTRONS, AND PHOTOELECTR
C
      DO 900 IDUM=1,2
      IPART=IDUM
      IF(IPART.EQ.1) TVOLTS= TVIONS
      IF(IPART.EQ.2) TVOLTS=-TVIONS/XMASS
      IF(IPART.EQ.1) SPEED0=SPEEDI
      IF(IPART.EQ.2) SPEED0=SPEEDE
      IF(IPART.EQ.1) TVCHG=TVIONS
      IF(IPART.EQ.2) TVCHG=-TVELEC
      IF(IPART.EQ.1) CURR=CURRI
      IF(IPART.EQ.2) CURR=CURRE
      IF(IPART.EQ.1) PARTCL(1)=PART1(1)
      IF(IPART.EQ.1) PARTCL(2)=PART1(2)
      IF(IPART.GT.1) PARTCL(1)=PART2(1)
      IF(IPART.GT.1) PARTCL(2)=PART2(2)
C
C*****
C     CALL DENSTY TO GENERATE NEW PARTICLES AND TAKE NEXT TIME STEP.
C*****
C
      IF(IPRINT.GT.1) WRITE(M,9750) PARTCL,SPEED0,CURR
```

```
C
      NINJ=NIINJ
      IF(IPART.EQ.2) NINJ=XIINJ*SPEEDE/SPEEDI+.1
C
      CALL DENSTY(INIT)
C
C     CDAMB=2.677*CENCC*SQRT(TVELEC)
C
      JVAR=MOD(JFIRST/2,10)+1
      DO 700 N=1,NTOT
      IF(IPART.EQ.2) GO TO 690
      Q(N)=XKOUN(N)*CHCON/XIINJ
      NETCH(N)=KOUNT(N)
      KHISTI(N,11)=KHISTI(N,11)+KOUNT(N)-KHISTI(N,JVAR)
      KHISTI(N,12)=KHISTI(N,12)+KOUNT(N)*KOUNT(N)-
     1 KHISTI(N,JVAR)*KHISTI(N,JVAR)
      KHISTI(N,JVAR)=KOUNT(N)
      GO TO 700
  690 CONTINUE
      Q(N)=Q(N)-XKOUN(N)*CHCON/XIINJ
      NETCH(N)=NETCH(N)-KOUNT(N)
      KHISTE(N,11)=KHISTE(N,11)+KOUNT(N)-KHISTE(N,JVAR)
      KHISTE(N,12)=KHISTE(N,12)+KOUNT(N)*KOUNT(N)-KHISTE(N,JVAR)*
     1 KHISTE(N,JVAR)
      KHISTE(N,JVAR)=KOUNT(N)
  700 CONTINUE
C
C ADD TO REMAINING VARIANCE SUMS
C
      IF(IPART.EQ.2) GO TO 720
      NIABS=NCABS
      NIACT=NCACT
      NIESC=NCESC
      KHABSI(11)=KHABSI(11)+NCABS-KHABSI(JVAR)
      KHACTI(11)=KHACTI(11)+NCACT-KHACTI(JVAR)
      KHABSI(12)=KHABSI(12)+NCABS*NCABS-KHABSI(JVAR)*KHABSI(JVAR)
      KHACTI(12)=KHACTI(12)+NCACT*NCACT-KHACTI(JVAR)*KHACTI(JVAR)
      KHABSI(JVAR)=NCABS
      KHACTI(JVAR)=NCACT
      GO TO 730
  720 CONTINUE
      NEABS=NCABS
      NEACT=NCACT
      NEESC=NCESC
      KHABSE(11)=KHABSE(11)+NCABS-KHABSE(JVAR)
      KHACTE(11)=KHACTE(11)+NCACT-KHACTE(JVAR)
      KHABSE(12)=KHABSE(12)+NCABS*NCABS-KHABSE(JVAR)*KHABSE(JVAR)
      KHACTE(12)=KHACTE(12)+NCACT*NCACT-KHACTE(JVAR)*KHACTE(JVAR)
      KHABSE(JVAR)=NCABS
      KHACTE(JVAR)=NCACT
  730 CONTINUE
```

```
C
      IFIRST=IFIRST+1
      JFIRST=JFIRST+1
  900 CONTINUE
      NCHGSE=0
      NCHGSI=0
      DO 710 J=1,JEDGE
      NCHGSE=NCHGSE+NCHGE(J)
      NCHGSI=NCHGSI+NCHGI(J)
  710 CONTINUE
C
      QDISK=FLOAT(NCHGSI-NCHGSE)*CHOON/XIINJ
C
      IF(IRELAX.EQ.0) GO TO 920
      DO 910 J=1,JEDGE
C
C     PHI RELATION WILL BE ESTABLISHED LATER.  THIS IS INCORRECT.
C
      PHI(J)=PI/2.*.9*QDISK/RADIUS
  910 CONTINUE
  920 CONTINUE
      N1=0
      N2=0
      IF(IPRINT.GT.0) WRITE(M,9740) (BLK,I,I=1,NR)
      DO 930 I=1,NZ
      N1=N2+1
      N2=N2+NR
      IF(IPRINT.GT.0) WRITE(M,9730) I,(NETCH(J),J=N1,N2)
  930 CONTINUE
C
C  BEGIN CALCULATION OF VARIANCES
C
      IF(ISW6.EQ.1.AND.JFIRST.EQ.2) CALL CONNEC(M)
      DELT=MIN0(JFIRST/2,10)
      AVIABS=FLOAT(KHABSI(11))/DELT
      AVIACT=FLOAT(KHACTI(11))/DELT
      RMIABS=SQRT(FLOAT(KHABSI(12))/DELT-AVIABS*AVIABS)
      RMIACT=SQRT(FLOAT(KHACTI(12))/DELT-AVIACT*AVIACT)
      AVEABS=FLOAT(KHABSE(11))/DELT
      AVEACT=FLOAT(KHACTE(11))/DELT
      RMEABS=SQRT(FLOAT(KHABSE(12))/DELT-AVEABS*AVEABS)
      RMEACT=SQRT(FLOAT(KHACTE(12))/DELT-AVEACT*AVEACT)
      IF(IPRINT.GT.0) WRITE(M,3002)
      IF(ISW6.EQ.1) CALL CONNEC(M)
 3002 FORMAT(///25X,21HSUMMARY FOR THIS STEP,16X,19HTEN-STEP RMS ERRORS,
     1 17X,17HTEN-STEP AVERAGES/22X,4HIONS,12X,9HELECTRONS,1X,2(3X,2(7X,
     2 1HI,7X,1HE))/1X,5HCYCLE,3X,4HTIME,2(4X,14HABS   ESC   ACT),
     3 2(8X,3HABS,5X,3HABS,5X,3HACT,5X,3HACT),2X,4HPMAX)
      IPLAST=IPRINT
      WRITE(M,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,
     1 RMIABS,RMEABS,RMIACT,RMEACT,AVIABS,AVEABS,AVIACT,AVEACT,XMAX
```

```
      WRITE(KV,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,
     1 RMIABS,RMEABS,RMIACT,RMEACT,AVIABS,AVEABS,AVIACT,AVEACT,XMAX
 3001 FORMAT(1X,I4,1PE10.3,1X,I3,2I6,3X,I3,2I6,3X,0P4F8.1,3X,4F8.1,F8.3)
      IF(ISW6.EQ.1) CALL DISCON(M)
CO
      IT=IT+1
      GO TO 650
C
 99   CONTINUE
C
C*****
C  TRANSFER SUMMARY FROM FILE=KV TO OUTPUT
C*****
      CALL DISCON(M)
      WRITE(M,5555)
      REWIND KV
      WRITE(M,3002)
  940 CONTINUE
      READ(KV,4444) LINE
      IF(EOF(KV)) 960,950
  950 CONTINUE
      WRITE(M,4444) LINE
      GO TO 940
  960 CONTINUE
C
C  CREATE RESTART FILE ON TAPE1
C
      IF(IV.EQ.2) GO TO 1010
      IV=1
      JV=2
      REWIND IV
      REWIND JV
C
C     COPY TAPE2 TO TAPE1
C
 1020 CONTINUE
      READ(JV) INJA,((VSAVE(I,J),I=1,10),J=1,INJA)
      IF(EOF(JV)) 1010,1030
 1030 CONTINUE
      WRITE(IV) INJA,((VSAVE(I,J),I=1,10),J=1,INJA)
      GO TO 1020
C
 1010 CONTINUE
C
C     CREATE RECORD TO BE COPIED TO INPUT FILE FOR RESTART
C
      REWIND JV
      WRITE(JV,9999) DATE
      WRITE(JV,1111) JEDGE,NR,NZA,NZB,IRELAX,MODE
      WRITE(JV,2222) (RR(J),J=1,NR)
      WRITE(JV,2222) (ZA(I),I=1,NZA)
```

```
      WRITE(JV,2222) (ZB(I),I=1,NZB)
      WRITE(JV,2222) (PHI(J),J=1,JEDGE)
      WRITE(JV,1111) NPRINT,NPTS,IT,ITS,NIINJ,ISKIP,JSKIP,MASS
      CALL RANGET(NRAN)
      WRITE(JV,3333) NRAN
      WRITE(JV,2222) DELTA,TIME,TVIONS,TVELEC,DENCC,WCLOUD
      WRITE(JV,1111) (NCHGI(I),I=1,JEDGE)
      WRITE(JV,1111) (NCHGE(I),I=1,JEDGE)
      WRITE(JV,2222) (Q(I),I=1,NTOT)
C
      STOP11
 1122 FORMAT(2I5,1P7E10.3/(  8E10.3))
 1111 FORMAT(16I5)
 2222 FORMAT(8E10.3)
 3333 FORMAT(1X,O20)
 4444 FORMAT(13A10,A6)
 5555 FORMAT(1H1)
 9990 FORMAT(//1X,I3,33H COLUMNS (R-VALUES) WITHIN RADIUS /
     1          1X,I3,25H COLUMNS (R-VALUES) TOTAL /
     2          1X,I3,46H ROWS (Z-VALUES) ABOVE AND INCLUDING Z=0 PLANE /
     3          1X,I3,46H ROWS (Z-VALUES) BELOW AND INCLUDING Z=0 PLANE /
     5          1X,I3,44H = IRELAX (RELAXATION,  =0 IF BODY POTENTIAL,
     6 10H IS FIXED,,48H  =1 IF BODY POTENTIAL IS FLOATING, AND EQUILIBR
     7 31HIUM VALUE IS TO BE CALCULATED.))
 9991 FORMAT(/1X,I3,41H = MODE,  SELECTS NORMAL CALCULATION MODE)
 9992 FORMAT(/1X,I3,29H = MODE,  SELECTS CLOUD MODEL)
 9993 FORMAT(/1X,I3,41H = MODE,  SELECTS WEIGHTED PARTICAL MODEL)
 9980 FORMAT(//24H R-VALUES WITHIN RADIUS=1PE15.4,3H CM/(1X,
     1 I3,1PE15.4,3H CM))
 9970 FORMAT(//24H R-VALUES OUTSIDE RADIUS/(1X,I3,1PE15.4,3H CM))
 9960 FORMAT(//34H Z-VALUES POSITIVE ABOVE Z=0 PLANE/(1X,I3,1PE15.4,
     1 3H CM))
 9950 FORMAT(//34H Z-VALUES NEGATIVE BELOW Z=0 PLANE/(1X,I3,1PE15.4,
     1 3H CM))
 9940 FORMAT(//27H SURFACE POTENTIALS ON BODY/(1X,I3,1PE15.4,
     1 6H VOLTS))
 9930 FORMAT(//38H INTEGER INPUTS FOR SHEATH CALCULATION /1X,
     1 I3,37H = NPRINT (TO PRINT TRAJECTORY STEPS)/1X,
     2 I3,39H = NPTS (TO COMPUTE ONE OR MORE POINTS)/1X,
     4 2I5,44H = IT AND ITS (ITERATION INDEX AND NUMBER OF,
     5 12H ITERATIONS)/1X,
     6 I5,30H  IONS GENERATED PER TIME STEP/1X,
     7 I5,35H  PAGES OF SUMMARY PRINTED FOR EACH,I5,6H STEPS,
     8 I3,30H = MASS (MASS OF COMPUTER ION))
 9920 FORMAT(//27H EMISSION AND PLASMA INPUTS /
     1 1X,F10.3,38H = DELTA  = INITIAL TIME STEP    (MEANS,
     2 1PE15.4,3X,8HSECONDS)/
     3 1X,0PF10.3,24H = TIME    = INITIAL TIME/
     4 1X,F10.3,49H = TVIONS = AMBIENT-ION      TEMPERATURE IN VOLTS/
     5 1X,F10.3,49H = TVELEC = AMBIENT-ELECTRON TEMPERATURE IN VOLTS/
     6 1X,F10.3,50H = DENCC  = AMBIENT-ELECTRON DENSITY IN NO. PER CC /
```

```
      7 20X,25H(ION PLASMA PERIOD      =,1PE15.4,3X,8HSECONDS)/
      8 20X,25H(ELECTRON PLASMA PERIOD =,E15.4,3X,8HSECONDS)/
      9 20X,25H(ELECTRON DEBYE LENGTH  =,E15.4,2X,3HCM)/
      A 1X,1PE15.4,50H = PCON    = NUMBER OF REAL IONS INJECTED PER CYCLE/
      B 1X,E15.4,53H = COMPAR = NUMBER OF REAL IONS/ELECTRONS PER COMPUTE
      C 10HR PARTICLE)
 9921 FORMAT(1X,1PE15.4,29H = WCLOUD  = CLOUD HALF WIDTH)
 9880  FORMAT(//1X,29H INPUTS FOR SINGLE TRAJECTORY/
      1 1X,F10.3,19H = RPT = R-POSITION/1X,F10.3,19H = ZPT = Z-POSITION/
      2 1X,F10.3,30H = AL1 = POLAR ANGLE (DEGREES)/
      3 1X,F10.3,34H = BE1 = AZIMUTHAL ANGLE (DEGREES)/
      4 1X,F10.3,24H = EV  = ENERGY IN VOLTS)
 9870  FORMAT(//13H ALL Z-VALUES/(1X,I3,1PE15.4))
 9860  FORMAT(//1X,21HINTERSTITIAL R-VALUES/(1X,I3,1PE15.4))
 9850  FORMAT(//1X,30HINTERSTITIAL Z-VALUES POSITIVE/(1X,I3,1PE15.4))
 9840  FORMAT(//1X,30HINTERSTITIAL Z-VALUES NEGATIVE/(1X,I3,1PE15.4))
 9835 FORMAT(//1X,43HINTERSTITIAL Z-VALUES NEGATIVE AND POSITIVE/
      1 (1X,I3,1PE15.4))
 9830  FORMAT(1H1/6X,1HN,5X,4HR(N),4X,4HZ(N)//)
 9790  FORMAT(/1X,I3,16H ORDER POTENTIAL)
 9780  FORMAT(/1X,I3,25H ORDER FLUXES AND CHARGES)
 9750  FORMAT(/1X,23HGOING INTO DENSTY WITH ,2A5,5X,
      1 13H WITH SPEED =,1PE15.4,7H CM/SEC,5X,22H AND CURRENT DENSITY =,
      2 E15.4,14H PICOAMP/CM**2)
 9740 FORMAT(//5X27HSUMMARY OF NET SPACE CHARGE//12X,9(A2,
      1 2HR(,I1,1H)),11(A1,2HR(,I2,1H)))
 9730 FORMAT(/5X,2HZ(,I2,1H),20I6)
      END
```

```
      SUBROUTINE LIST(LST,IP)
C
C     PRINT ARRAYS
C
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 NONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51),
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      DIMENSION KOUT(5),ROUT(5),ZOUT(5)
C
      DO 500 LINE=1,60
      DO 200 NP=1,5
      KP=LINE + (NP-1)*60 + (IP-1)*300
      IF(KP.GT.NTOT.AND.NP.EQ.1) RETURN
      IF(KP.GT.NTOT) GO TO 300
      NMAX=NP
      KOUT(NP)=KP
      IKP=(KP-1)/NR+1
      JKP=MOD(KP-1,NR)+1
C
      IF(LST.EQ.1) ROUT(NP)=Q(KP)
      IF(LST.EQ.1) ZOUT(NP)=X(KP)
      IF(LST.EQ.2) ROUT(NP)=RR(JKP)
      IF(LST.EQ.2) ZOUT(NP)=ZZ(IKP)
C
200   CONTINUE
C
 300  CONTINUE
      IF(LST.NE.1) GO TO 450
C$1300    GO TO (400,450),LST
C
400   WRITE(M,1000) (KOUT(NP),ROUT(NP),ZOUT(NP),NP=1,NMAX)
1000  FORMAT(5(I5,1P2E11.3))
      GO TO 500
C
450   WRITE(M,3000) (KOUT(NP),ROUT(NP),ZOUT(NP),NP=1,NMAX)
3000  FORMAT(5(I8,2F8.2))
500   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIELD
C
C     COMPUTE ELECTRIC FIELD FROM GIVEN FIXED POTENTIALS OR CHARGES ON S
C     CONSTRUCT COEFFICIENTS IN LINEAR DIFFERENCE EQUATIONS. SOLVE BY RE
C
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 MONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51),
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      COMMON/FLD/DEBYE,DIELEC,IRELAX,PHI(11),DENCC
      COMMON/ARRAYC/ INDX(51,4),COEF(51,5)
      COMMON/CCC/N,I,J,NN,NS,NE,NW,NSEW,CN,CS,CE,CW,CC,C,AREA,R,Z
     1 ,VOLUME,CHARG
      REAL KKK1,KKK2
      INTEGER DIA
      DATA KKK1/5HALPH=/
      DATA KKK2/5HBET =/
      DATA CN/5HNORTH/
      DATA CS/5HSOUTH/
      DATA CE/5HEAST /
      DATA CW/5HWEST /
C
C     ASSUME ASYMPTOTIC BEHAVIOR AT INFINITY (MONOPOLE+DIPOLE,ETC.)
C
      ALF(RRR,ZZZ)=ABS(((A+3.*B*ZZZ/RS)*ZZZ/RS-B/RS)/(A+B*ZZZ/RS))
C
      BEF(RRR,ZZZ)= ABS((A+3.*B*ZZZ/RS)*RRR/RS/(A+B*ZZZ/RS))
C
C     DIA=POSITIVE INTEGER FOR DIAGNOSTIC OUTPUT
      DIA=1
      DIA=0
      IF(JFIRST.EQ.0) WRITE(M,9000) IT,TIME
 9000 FORMAT(1H1/10H0FIELD CALCULATION,5X,4HIT =,I3,5X,9HAT TIME =,
     1 1PE10.3////1X,17HCOEFFICIENT ARRAY)
C
C     PURE DIPOLE
      A=0.
      B=1.
C
C     PURE MONOPOLE
      A=1.
      B=0.
C
      ZOLD=ZZ(1)
      N=0
      DO 1000 IDM=1,NZ
      DO 1000 JDM=1,NR
      N=N+1
      R=RR(JDM)
      Z=ZZ(IDM)
      RS=R**2 + Z**2
      IF(IDM.LT.NZA) VOLUME=AREAH(JDM)*DELZ(IDM)
```

```
      IF(IDM.GT.NZA) VOLUME=AREAH(JDM)*DELZ(IDM+1)
      IF(IDM.EQ.NZA) VOLUME=AREAH(JDM)*(DELZ(NZA)+DELZ(NZA+1))
      CHARG=Q(N)
      IF(Z.GE.ZOLD.AND.N.GT.1) GO TO 100
      ZOLD=Z
      IF(JFIRST.EQ.0) WRITE(M,8000) IDM,Z
8000  FORMAT(//1X,2HZ(,I2,2H)=,F10.3/
     1 20X,1HN,17X,1HW,17X,1HC,17X,1HE,17X,1HS,15X,6HCHARGE,
     2 4X,13HNORMAL CHARGE)
C
100   CC=0.
      I=IDM
      J=JDM
      DO 200 JDUM=1,2
      NSEW=JDUM
      C=0.
      CALL CNSEW(1)
C
      IF(NSEW.EQ.1) OO=ON
      IF(NSEW.EQ.2) OO=OS
      IF(DIA.GT.0) WRITE(M,8888) N,I,J,OO,AREA,C
8888  FORMAT(1X,16HN,I,J,OO,AREA,C=,I4,2X,2I3,1X,A5,1P2E16.4)
C
      CC=CC+C
      IF(C.GT.0.) GO TO 150
C
      ALPH=0.
      IF(Z.EQ.ZZ(1).OR.Z.EQ.ZZ(NZ)) ALPH=ALF(R,Z)
      ALBE=KKK1
C
      IF(DIA.GT.0) WRITE(M,7777) ALBE,N,I,J,OO,AREA,ALPH
7777  FORMAT(1X,14HN,I,J,OO,AREA,,A5,I4,2X,2I3,1X,A5,1P2E16.4)
C
      IF(ALPH.GT.0.) CC=CC + AREA*ALPH
C
150   IF(NSEW.EQ.1) CN=C
      IF(NSEW.EQ.2) CS=C
200   CONTINUE
C
      DO 300 JDUM=1,2
      NSEW=JDUM
      C=0.
      CALL CNSEW(2)
      IF(NSEW.EQ.1) OO=OE
      IF(NSEW.EQ.2) OO=OW
      IF(DIA.GT.0) WRITE(M,8888) N,I,J,OO,AREA,C
C
      CC=CC+C
      IF(C.GT.0.) GO TO 350
C
      BET=0.
```

```
      IF(R.EQ.RR(NR)) BET=BEF(R,Z)
      ALBE=KKK2
C
      IF(DIA.GT.0) WRITE(W,7777) ALBE,N,I,J,OO,AREA,BET
      IF(BET.GT.0.) CC=CC + AREA*BET
C
350   IF(NSEW.EQ.1) CE=C
      IF(NSEW.EQ.2) CW=C
300   CONTINUE
      CALL ARRAYS
1000  CONTINUE
C
      CALL RELAX
      RETURN
      END
```

```
      SUBROUTINE CNSEW(IGO)
C
C     COEFFICIENTS NORTH AND SOUTH
C
      COMMON M,IW,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
    1 MONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51),
    2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      COMMON/FLD/DESVE,DIELEC,IRELAX,PHI(11),DENCC
      COMMON/CCC/N,I,J,NN,NS,NE,NW,NSEW,CN,CS,CE,CW,CC,C,AREA,R,Z
    1 ,VOLUME,CHARG
      NF(IZ,JR)=JR + NR*(IZ-1)
C
      IF(IGO.GT.1) GO TO 550
C
      AREA=0.
      DZ=0.
      IF(Z.EQ.0.) GO TO 100
      IF(Z.GT.0.) AREA=AREAH(J)
      IF(Z.LT.0.) AREA=DIELEC*AREAH(J)
      GO TO 200
C
100   IF(NSEW.EQ.1) AREA=AREAH(J)
      IF(NSEW.EQ.2) AREA=DIELEC*AREAH(J)
C
200   IF(NSEW.EQ.1) GO TO 300
      IF(NSEW.EQ.2) GO TO 400
      RETURN
C
300   NN=0
      IF(I.EQ.1) RETURN
      NN=NF(I-1,J)
      DZ=ZZ(I-1) - Z
      GO TO 500
C
400   NS=0
      IF(I.EQ.NZ) RETURN
      NS=NF(I+1,J)
      DZ=Z - ZZ(I+1)
C
500   IF(DZ.GT.0.) C=AREA/DZ
      RETURN
C
C
550   CONTINUE
C
      AREA=0.
      DR=0.
      IF(Z.EQ.0.) AREA=DELZ(NZA) + DIELEC*DELZ(NZA+1)
      IF(Z.GT.0.) AREA=DELZ(I)
      IF(Z.LT.0.) AREA=DIELEC*DELZ(I+1)
C
```

```
      IF(NSEW.EQ.1) GO TO 600
      IF(NSEW.EQ.2) GO TO 700
      RETURN
C
600   NE=0
      IF(J.EQ.NR)  AREA=2.*PI*RR(J)*AREA
      IF(J.EQ.NR) RETURN
      NE=NF(I,J+1)
      DR=RR(J+1) - R
      PERIM=PI*(RR(J+1) + R)
      GO TO 800
C
700   NW=0
      IF(J.EQ.1) AREA=0.
      IF(J.EQ.1) RETURN
      NW=NF(I,J-1)
      DR=R - RR(J-1)
      PERIM=PI*(R + RR(J-1))
C
800   AREA=PERIM*AREA
      IF(DR.GT.0.) C=AREA/DR
      RETURN
      END
```

```
      SUBROUTINE ARRAYS
C
C     SAVE COEFFICIENTS AND INDICES IN ARRAYS
C
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 NONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51),
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      COMMON/FLD/DEBYE,DIELEC,IRELAX,PHI(11),DENCC
      COMMON/ARRAYC/ INDX(51,4),COEF(51,5)
      COMMON/CCC/N,I,J,NN,NS,NE,NW,NSEW,CN,CS,CE,CW,CC,C,AREA,R,Z
     1 ,VOLUME,CHARG
C
C
      IF(CC.EQ.0.) CALL ABNOR(10H   STOP111)
      IF(CC.EQ.0.) STOP111
C     CN=NORTH (=+Z) NEIGHBOR = COEF1*CC
C
      COEF(N,1)=CN/CC
C
C     CS=SOUTH (=-Z) NEIGHBOR = COEF2*CC
C
      COEF(N,2)=CS/CC
C
C     CE=EAST (=+R) NEIGHBOR = COEF3*CC
C
      COEF(N,3)=CE/CC
C
C     CW=WEST (=-R) NEIGHBOR = COEF4*CC
C
      COEF(N,4)=CW/CC
C
C     WHERE CC=CENTRAL-POINT COEFFICIENT
C
      COEF(N,5)= .9*4.*PI*CHARG/CC
C58   IF(DEBYE.GT.0.) COEF(N,5)=CH(N)/CC/DEBYE**2*.9
C
C     CH=NET POSITIVE CHARGE IN N-TH BOX (FACTOR .9 FOR CH IN PICOCOULOM
C     POTENTIAL IN VOLTS, LENGTH IN CM)
C
      INDX(N,1)=NN
      INDX(N,2)=NS
      INDX(N,3)=NE
      INDX(N,4)=NW
      CHARG0=1.6E-7*VOLUME*DENCC
C
C
      IF(JFIRST.EQ.0) WRITE(M,1000) NN,CN,NW,CW,N,CC,NE,CE,NS,CS,CHARG,
     1 CHARG0
 1000 FORMAT(/13X,5(1X,1H(,I4,2H)=,1PE10.4),2E14.4)
      RETURN
      END
```

```
      SUBROUTINE RELAX
C
C     POINT-SUCCESSIVE RELAXATION TO SOLVE EQUATIONS
C
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 MONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),X(51),Q(51),
     2 AREAH(11),OELZ(11),RI(11),ZCI(11)
      COMMON/FLD/DEBYE,DIELEC,IRELAX,PHI(11),DENCC
      COMMON/ARRAYC/ INOX(51,4),COEF(51,5)
C
      OMEGA=1.9
      EPS=1.E-9
      ITMAX=2000
      ITR=0
      IPROLD=0
      IGO=1
      IF(JFIRST.GT.0.AND.IT.GT.0) GO TO 200
      DO 100 K=1,NTOT
100   X(K)=0.
C
200   CONTINUE
      ITR=ITR+1
      DELTAM=0.
C
      DO 500 N=1,NTOT
      X1=X(N)
      I=(N-1)/NR+1
      J=MOD(N-1,NR)+1
      IF(J.GT.JEDGE.OR.ZZ(I).NE.0.) GO TO 400
C
C  SET SURFACE POTENTIALS
C
      X(N)=PHI(J)
      GO TO 500
 400  CONTINUE
C
      SUM=COEF(N,5)
      DO 300 KK=1,4
      INDEX=INDX(N,KK)
      IF(INDEX.GT.0) SUM=SUM + COEF(N,KK)*X(INDEX)
300   CONTINUE
C
      X(N)=SUM
      X(N)=OMEGA*X(N) + (1.-OMEGA)*X1
      DELTA=ABS(X(N)-X1)
      IF(X1.NE.0.) DELTA=ABS((X(N)-X1)/X1)
      IF(DELTA.GT.DELTAM) DELTAM=DELTA
C
500   CONTINUE
C
      IF(ITR.GT.ITMAX) WRITE(M,8888) ITR
```

```
      IF(ITR.GT.ITMAX) GO TO 700
 8888 FORMAT(////10H MORE THAN,I4,11H ITERATIONS)
C
      IPR=ITR/500
      IF(IPR.LE.IPROLD) GO TO 600
      IPROLD=IPR
      GO TO 800
C
  600 IF(DELTAM.GT.EPS) GO TO 200
C
C     ITERATION CONVERGED.  PRINT AND EXIT.
C
  700 IGO=2
      IF(IPRINT.EQ.0) GO TO 1000
  800 NFPP=(NTOT/300) + 1
      DO 900 IPAGE=1,NFPP
      WRITE(M,7777) IT,TIME,ITR,EPS,DELTAM,OMEGA
      CALL LIST(1,IPAGE)
  900 CONTINUE
 7777 FORMAT(1H1/18H0FIELD CALCULATION,5X,4HIT =,I3,5X,9HAT TIME =,
     1 1PE10.3//15H SOLUTION AFTER,I6,2X,25HITERATIONS WITH TOLERANCE,
     2 0PF12.8,8X,18HMAXIMUM DIFFERENCE,F12.8,6X,6HOMEGA=,F8.5/4X,
     3 1HN,2X,4HQ(N),7X,4HX(N)//)
C
      IF(IGO.EQ.1) GO TO 600
 1000 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTERP(R,Z,JG,IG,PHIC,RI,ZI,IR,IZ,INT)
C
C     INTERPOLATE IN R-Z GRID TO FIND POTENTIAL AND GRADIENT
C
      COMMON/TT/MINT,X,Y,XDOT,YDOT,ZDOT,PHI,PHIR,PHIZ,DT
      DIMENSION ZI(11),RI(11),PHIC(IR,IZ)
      IF(INT.EQ.0) IG=0
      IF(INT.EQ.0) JG=0
      NCH=0
C     LOCATION OF Z IN ARRAY
C
C     ASSUME ZI(IZ) LESS THAN OR EQUAL TO Z LESS THAN OR EQUAL TO ZI(1).
C
      IF(Z.EQ.ZI(1)) IG=2
      IF(Z.EQ.ZI(1)) GO TO 103
      IF(INT.EQ.1) GO TO 100
C
      DO 10 I=2,IZ
      IG=IZ-I+2
      IF(Z.LT.ZI(IG-1)) GO TO 103
10    CONTINUE
      GO TO 999
C     RETURN WITHOUT LOCATING Z
C
C     ACCEPT IF ZI(IG) LESS THAN OR EQUAL TO Z LESS THAN ZI(IG-1).
C
100   IF(Z.GE.ZI(IG-1)) GO TO 102
C     IG TOO LARGE.  USE DECREMENT LOOP
C
      IF(Z.GE.ZI(IG)  ) GO TO 104
101   IG=IG+1
      IF(Z.LT.ZI(IG)  ) GO TO 101
      GO TO 103
102   IG=IG-1
      IF(Z.GE.ZI(IG-1)) GO TO 102
103   NCH=1
C     IG DETERMINED
C
104   CONTINUE
C
C     LOCATION OF R IN ARRAY
C
C     ASSUME RI(1) LESS THAN OR EQUAL TO R LESS THAN OR EQUAL TO RI(IR).
C
      IF(R.EQ.RI(IR)) JG=IR-1
      IF(R.EQ.RI(IR)) GO TO 153
      IF(INT.EQ.1) GO TO 150
C
      DO 15 J=2,IR
      JG=J-1
      IF(R.LT.RI(J)) GO TO 153
```

```
15      CONTINUE
        GO TO 999
C       RETURN WITHOUT LOCATING R
C
C       ACCEPT IF RI(JG) LESS THAN OR EQUAL TO R LESS THAN RI(JG+1)
C
150     IF(R.GE.RI(JG+1)) GO TO 152
        IF(R.GE.RI(JG)  ) GO TO 154
151     JG=JG-1
        IF(R.LT.RI(JG)  ) GO TO 151
        GO TO 153
152     JG=JG+1
        IF(R.GE.RI(JG+1)) GO TO 152
153     NCH=1
154     CONTINUE
C
        IF(IG.LE.0.OR.JG.LE.0) GO TO 999
C
C       RETURN IF IG AND JG AND ALL THAT WERE REQUESTED
C
        IF(INT.EQ.2) GO TO 310
C
C       SET UP FRACTIONS
C
        DELZ=ZI(IG-1)-ZI(IG)
        DELR=RI(JG+1)-RI(JG)
        FZ=(Z-ZI(IG))/DELZ
        FR=(R-RI(JG))/DELR
C
C       DEFINE POTENTIALS AT CORNERS OF BOX
C
        P22=PHIC(JG+1,IG-1)
        P21=PHIC(JG,  IG-1)
        P12=PHIC(JG+1,IG  )
        P11=PHIC(JG,  IG  )
        IF(NCH.EQ.0) GO TO 300
C
C       SKIP IF PHI-BOX IS NOT CHANGED
C
        D1=(P22-P12)/DELZ
        D2=(P21-P11)/DELZ
        D3=(P22-P21)/DELR
        D4=(P12-P11)/DELR
C
300     CONTINUE
C
C       INTERPOLATE TO FIND POTENTIAL AND COMPONENTS OF GRADIENT
C
        PHIZ=D2 + FR*(D1-D2)
        PHIR=D4 + FZ*(D3-D4)
        PHI=P11 + FR*(P12-P11) + FZ*(P21-P11) + FR*FZ*(P22-P21-P12+P11)
```

```
  310  CONTINUE
       RETURN
C
C
C      CANNOT LOCATE EITHER R OR Z OR BOTH
C
999    WRITE(MINT,9999) R,Z,IG,JG
       CALL ABNOR(10H     STOP22)
       STOP22
C
9999   FORMAT(///1X,32HCANNOT LOCATE R OR Z. R,Z,IG,JG=,1P2E12.4,2I5)
       END
```

```
      SUBROUTINE TRACK(R,Z)
C
C     ADVANCE X,Y,Z,XDOT,YDOT,ZDOT BY CONSTANT ACCELERATION. WITH STEP C
C
      COMMON/TT/MINT,X,Y,XDOT,YDOT,ZDOT,PHI,PHIR,PHIZ,DT
C
      IF(R.EQ.0.) PHIX=0.
      IF(R.EQ.0.) PHIY=0.
      IF(R.EQ.0.) GO TO 100
C
      PHIX=PHIR*X/R
      PHIY=PHIR*Y/R
C
  100 CONTINUE
C
      X=X + DT*(XDOT - .25*DT*PHIX)
      Y=Y + DT*(YDOT - .25*DT*PHIY)
      Z=Z + DT*(ZDOT - .25*DT*PHIZ)
C
      XDOT=XDOT - .5*DT*PHIX
      YDOT=YDOT - .5*DT*PHIY
      ZDOT=ZDOT - .5*DT*PHIZ
C
      RETURN
      END
```

```
      FUNCTION ERF(X,ERROR)
C
C     ERROR FUNCTICN
C
      DIMENSION A(7)
      DATA A,RPI /1.,0.70523078E-1,.42282012E-1,.92705272E-2,
     1 .15201430E-3,.27656720E-3,.430638E-4,1.7724538/
      S=X*X
      RS=ABS(X)
      IF(S.LT.4.0) GO TO 240
      S=AMIN1(S,675.)
      ERROR=1./(RS*RPI)*(1.+.5/S*(-1.+.5/S*(3.-7.5/S)))
      ERF=1.-ERROR*EXP(-S)
      RETURN
C
C     HASTINGS APPROXIMATION  (P.187)
C
  240 SP=1.0
      SM=1.0
      DO 300 I=2,7
      SP=SP*RS
      SM=SM+A(I)*SP
      IF(ABS(SP).LT.1.E-9) GO TO 350
  300 CONTINUE
  350 ERFC=1./SM**16
      ERF=1.-ERFC
      RETURN
      END
```

```
      FUNCTION ERFINV(R)
C
C     INVERSE ERF RATIONAL APPROXIMATION
C     (SEE ABRAMOWITZ & STEGUN, P. 933)
C
      DATA C0,C1,C2/2.515517,.802853,.010328/
      DATA D1,D2,D3/1.432788,.189269,.001308/
      DATA ROUND,ROOT2,ROOTPI/1.E-10,1.4142135623730,1.77245385/
      ERFINV=-2.
      IF(R.GT.1..OR.R.LT.0.) CALL ABNOR(10H    STOP 33)
      IF(R.GT.1..OR.R.LT.0.) STOP 33
      X=0.
      IF(R.LE..09) X=R*ROOTPI/2.
      IF(R.LE..09) GO TO 200
      P=(1.-R)/2.
      IF(P.LE.0.) GO TO 200
      T=0.
      RADIC=0.
      IF(P.LE.ROUND) GO TO 150
      IF(P.GT.ROUND) RADIC=-2.*ALOG(P)
      IF(RADIC.GT.0.) T=SQRT(RADIC)
      XN=C0+T*(C1+T*C2)
      XD=1.+T*(D1+T*(D2+T*D3))
      XP=T
      IF(XD.GT.0.) XP=XP-XN/XD
      X=XP/ROOT2
      IF(X.LT.ROUND) X=0.
      GO TO 200
  150 CONTINUE
      IF(P.GT.0.) T=ALOG(.5/ROOTPI/P)
      IF(T.GT.0.) RADIC=T - .5*ALOG(T)
      IF(RADIC.GT.0.) X=SQRT(RADIC)
  200 CONTINUE
      ERFINV=X
      END
```

```
      SUBROUTINE PFLOT(U,RI,ZI,NRI,NZI,NRO,NZO,RADIUS,XMAX,IT)
      DIMENSION ILINE(122),JLINE(122),BNDY(23),RI(11),ZI(11),
     1 REXP(122),ISYM(26),U(NRI,NZI)
      COMMON /IT/ MINT,X,Y,XDOT,YDOT,ZDOT,PHI,PHIR,FFIZ,DT
      DATA ISYM /1HZ,1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL,
     1 1HM,1HN,1HO,1HP,1HQ,1HR,1HS,1HT,1HU,1HV,1HW,1HX,1HY/
      DATA BNDY /100.,10.,9.,8.,7.,6.,5.,4.,3.,2.,1.,0.,-1.,-2.,-3.,
     1 -4.,-5.,-6.,-7.,-8.,-9.,-10.,-100./
      MINT=6
C
******
C  DETERMINE MINIMUM AND MAXIMUM OF THE X MATRIX
C*****
      XMIN=10.**10
      XMAX=-XMIN
      NROM=NRO-1
      NZOM=NZO-1
      DO 100 JJ=1,NRI
      DO 100 II=1,NZI
      XMAX=AMAX1(XMAX,U(JJ,II))
      XMIN=AMIN1(XMIN,U(JJ,II))
  100 CONTINUE
C
C*****
C  LIST SYMBOL CORRESPONDENCE
C*****
      WRITE(MINT,3001) XMAX,XMIN,IT
 3001 FORMAT(1H1,5X,*SYMBOL TABLE CORRESPONDENCE FOR PRINTER PLOT*/
     1 6X,*TWENTY TWO INTERVALS ASSUMED FOR DATA.     XMAX = *,
     2 1PE12.4,*  TO XMIN = *,E12.4,*  AT BEGINNING OF CYCLE IT =*,
     3 I6// 24X,*INTERVAL*/8X,*SYMBOL*,5X,*FROM*,12X,*TO*)
      WRITE(MINT,3002) (I,ISYM(I),BNDY(I),BNDY(I+1),I=1,22)
 3002 FORMAT(5X,I3,2X,A1,1P2E15.5)
      WRITE(MINT,3003)
 3003 FORMAT(1H1//)
C
C*****
C  EXPAND R COORD AND PREPARE TO INTERPOLATE
C*****
      REXP(1)=RI(1)
      DRR=(RI(NRI)-RI(1))/FLOAT(NROM)
      DO 200 II=2,NRO
      REXP(II)=REXP(II-1)+DRR
  200 CONTINUE
      Y=0.
      DZZ=(ZI(1)-ZI(NZI))/FLOAT(NZOM)-1.E-10
      Z=ZI(1)+DZZ
      IZCUR=1
      DO 350 KK=1,NZO
      Z=Z-DZZ
      IFLG=0
```

```
      IF(Z.LE.ZI(IZCUR)) IFLG=1
      JRCUR=1
      INT=0
      DO 300 II=1,NRO
      X=R=REXP(II)
      JFLG=0
      IF(X.GE.RI(JRCUR)) JFLG=1
      CALL INTERP(R,Z,JG,IG,U,RI,ZI,NRI,NZI,INT)
      INT=1
C
C     DETERMINE BOX NUMBER FROM BNDY MATRIX
C
      DO 250 JJ=1,22
      IBOX=JJ
      TBOX=((BNDY(JJ)-PHI)*(PHI-BNDY(JJ+1)))
      IF(IBOX.EQ.22) ILINE(II)=ISYM(22)
      IF(TBOX.GT.0.) GO TO 750
  250 CONTINUE
  750 CONTINUE
      ILINE(II)=1H
      IF(MOD(IBOX,2).EQ.1) ILINE(II)=ISYM(MOD(IBOX-1,26)+1)
C
C  MARK CORNERS
C
      IF(IFLG.EQ.1.AND.JFLG.EQ.1) ILINE(II)=1H+
C
C  MARK DISC
C
      IF(ZI(IZCUR).EQ.0..A.REXP(II).LE.RADIUS.A.IFLG.EQ.1) ILINE(II)=1H-
      IF(JFLG.EQ.1) JRCUR=JRCUR+1
C
  300 CONTINUE
C
C*****
C  HAVE WE CROSSED A Z BOUNDARY?????
C*****
      IF(IFLG.EQ.1) GO TO 310
C
C  ELSE
      WRITE(MINT,3004) (ILINE(I),I=1,NRO)
 3004 FORMAT(9X,122A1)
      GO TO 350
C
  310 CONTINUE
      WRITE(MINT,3005) ZI(IZCUR),(ILINE(I),I=1,NRO)
 3005 FORMAT(1X,F7.2,1X,80A1)
      IZCUR=IZCUR+1
  350 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DENSTY(INIT)
C
C     COMPUTE FLUXES AND CHARGE DENSITIES
C
      COMMON M,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 MONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),U(51),Q(51),
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      COMMON/DEN/RPT,ZPT,AL1,BE1,EV,TVOLTS,DENST,NPTS,SPEED0,CURR,
     1 IPART,PARTCL(2),DELTA,SPEED,QDISK,DZMIN,DTSEC,NINJ,
     2 VSAVE(10,100),TVCHG,NRAN,NCESC,NCABS,NCACT,NCHGE(11),NCHGI(11),
     3 KOUNT(51),XKOUN(51),MODE,NCLOUD
      COMMON/TT/MINT,X,Y,XDOT,YDOT,ZDOT,PHI,PHIR,PHIZ,DT
      DIMENSION ENCESC(2),ENCABS(2),PHIC(51),WT(51),PHYR(51),PHYZ(51),
     1 PHYALT(11),IFDSK(11)
      DATA ENDESC/5H ESCA,5HPES    /,ENDABS/5HABSOR,5HBED    /
      NIJ(I,J)=NR*(I-1)+J
      IF(IPRINT.GT.0) WRITE(M,1000) IT,PARTCL,SPEED0,CURR
C
      JV=3-IV
      IF(IFIRST.NE.0) GO TO 90
      CALL RANSET(NRAN)
      NPARTS=0
      HITE=ZZ(1)-ZZ(NZ)
      AREAA=PI/2.*RR(NR)*RR(NR)
      AREAB=RR(NR)*HITE
      ROOTPI=SQRT(PI)
      NTOP=0
      NBOT=0
      NSID=0
      IC=1
      JG=1
      MINT=M
      N1=1
      INJE=0
      BLK=1H
      REWIND IV
      REWIND JV
C
      IF(IT.NE.0) GO TO 60
C
      DO 80 I=1,JEDGE
      NCHGE(I)=0
      NCHGI(I)=0
   80 CONTINUE
      GO TO 90
   60 CONTINUE
      READ(IV) INJA,((VSAVE(I,J),I=1,10),J=1,INJA)
      IF(EOF(IV)) 90,70
   70 CONTINUE
      WRITE(JV) INJA,((VSAVE(I,J),I=1,10),J=1,INJA)
      NPARTS=VSAVE(9,INJA)+.1
```

```
      GO TO 60
C
   90 CONTINUE
      IF(INIT.EQ.1) GO TO 350
      NCABS=0
      NCESC=0
      NCACT=0
      INTER=0
      N=0
      IF(IPRINT.GT.0.AND.IPART.EQ.2) WRITE(M,1010) (RR(J),J=1,NR)
      IF(IPRINT.GT.0.AND.IPART.EQ.1) WRITE(M,1020) (RR(J),J=1,NR)
      J1=1-NR
      J2=0
      DO 110 I=1,NZA
      J1=J1+NR
      J2=J2+NR
      DO 100 J=1,NR
      N=N+1
      IF(TVOLTS.EQ.0.) GO TO 110
  100 PHIC(N)=U(N)/TVOLTS
      IF(IPRINT.GT.0) WRITE(M,1100) I,ZZ(I),(PHIC(J),J=J1,J2)
  110 CONTINUE
C
      IF(IPRINT.GT.0) WRITE(M,1200)
      I1=NZA+1
      DO 160 I=I1,NZ
      J1=J1+NR
      J2=J2+NR
      DO 150 J=1,NR
      N=N+1
      IF(TVOLTS.EQ.0) GO TO 160
  150 PHIC(N)=U(N)/TVOLTS
      IF(IPRINT.GT.0) WRITE(M,1100) I,ZZ(I),(PHIC(J),J=J1,J2)
  160 CONTINUE
      IF(MODE.NE.1) GO TO 180
C
C  CALCULATE PHYR AND PHYZ MATRICES
C
      N=0
      DO 170 I=1,NZ
      IP=MIN0(I+1,NZ)
      IM=MAX0(I-1,1)
C
      DO 170 J=1,NR
      N=N+1
      JP=MIN0(J+1,NR)
      JM=MAX0(J-1,1)
C
      NP=NIJ(I,JP)
      NM=NIJ(I,JM)
      PHYR(N)=(PHIC(NP)-PHIC(NM))/(RR(JP)-RR(JM))
```

```
      IF(I.EQ.NZA.AND.J.EQ.JEDGE+1) PHYR(N)=(PHIC(NP)-PHIC(NM))/
     1 (RR(JP)-RADIUS)
C
      NM=NIJ(IM,J)
      NP=NIJ(IP,J)
      PHYZ(N)=(PHIC(NM)-PHIC(NP))/(ZZ(IM)-ZZ(IP))
      IF(I.EQ.NZA.AND.J.LE.JEDGE) PHYZ(N)=(PHIC(NM)-PHIC(N))/
     1 (ZZ(IM)-ZZ(I))
      IF(I.EQ.NZA.AND.J.LE.JEDGE) PHYALT(J)=(PHIC(N)-PHIC(NP))/
     1 (ZZ(I)-ZZ(IP))
  170 CONTINUE
  180 CONTINUE
C
      IF(NPTS.NE.0) GO TO 350
C
C     SINGLE TRAJECTORY  (NPTS=0)
C
      ALPHA=AL1*PI/180.
      BETA=BE1*PI/180.
      WRITE(M,1300) AL1,BE1,ALPHA,BETA,EV
C$1   GO TO 400
C*****
C  17 AUG 78.  NOTE THAT SINGLE TRAJECTORY IS INCOMPLETE
C     BEGINNING HERE.  TEMPORARILY THIS CASE IS JUST TERMINATED.
C*****
      CALL ABNOR(10H   STOP400)
      STOP 400
C     INITIAL CONDITIONS OF TRAJECTORIES
C
 350  CONTINUE
C*****
C  INJECT NEW PARTICLES
C  ASSUMED FLUX = NINJ IONS PER CYCLE
C*****
C
      DO 370 I=1,NINJ
      RAN1=RANF(NRAN)
      RAN2=RANF(NRAN)
      RAN3=RANF(NRAN)
      RAN4=RANF(NRAN)
      RAN5=RANF(NRAN)
      RAN6=RANF(NRAN)
      NPARTS=NPARTS+1
      INJ0=INJ0+1
C
C  SELECT VERTICAL AND PERPENDICULAR COMPONENTS OF VELOCITY
C
      VPERP=SQRT(ABS(ALOG(RAN1)))
C
      VVERT=ERFINV(ABS(1.-2.*RAN2))
      IF(RAN2.GT..5) VVERT=-VVERT
```

```
C
C   CALCULATE THETA AND TRIG FUNCTIONS OF THETA
C
      AL1R=ATAN2(VPERP,VVERT)
      CAL1=COS(AL1R)
      SAL1=SIN(AL1R)
C
C   SELECT TOP/BOTTOM OR SIDE
C
      RAREAS=AREAA*ABS(CAL1)/(AREAB*SAL1+AREAA*ABS(CAL1))
      IF(RAN3.LE.RAREAS) GO TO 360
C
C     PARTICLE HITS SIDE.  CALCULATE X,Y,Z AND PHI COORD.
C
      LOCAT=4HSIDE
      SBE1=RAN4
      BE1R=ASIN(SBE1)
      CBE1=COS(BE1R)
      NSID=NSID+1
C
      X=RR(NR)
      IF(INIT.EQ.1) X=SQRT(RAN6)*RR(NR)
      Y=0.
      Z=RAN5*(HITE)+ZZ(NZ)
C
      GO TO 410
 360  CONTINUE
C
C     PARTICLE HITS TOP/BOTTOM.  CALCULATE X,Y,Z AND PHI COORD.
C
      BE1R=2.*PI*RAN4
      CBE1=COS(BE1R)
      SBE1=SIN(BE1R)
C
      X=SQRT(RAN5)*RR(NR)
      Y=0.
      Z=ZZ(NZ)
      IF(CAL1.LT.0.) Z=ZZ(1)
      IF(INIT.EQ.1) Z=HITE*RAN6+ZZ(NZ)
      LOCAT=6HBOTTOM
      IF(CAL1.LT.0.) LOCAT=3HTOP
      IF(CAL1.GE.0.) NBOT=NBOT+1
      IF(CAL1.LT.0.) NTOP=NTOP+1
 410  CONTINUE
C
C   CALCULATE VELOCITY COMPONENTS
C
      XDOT=-SPEED0*VPERP*CBE1
      YDOT=SPEED0*VPERP*SBE1
      ZDOT=SPEED0*VVERT
C
```

```
      VSAVE(1,INJB)=X
      VSAVE(2,INJB)=Y
      VSAVE(3,INJB)=Z
      VSAVE(4,INJB)=XDOT
      VSAVE(5,INJB)=YDOT
      VSAVE(6,INJB)=ZDOT
      VSAVE(7,INJB)=DELTA*RAN6
      IF(INIT.EQ.1) VSAVE(7,INJB)=DELTA
      IF(IPART.EQ.1) VSAVE(8,INJB)=1.
      IF(IPART.EQ.2) VSAVE(8,INJB)=-1.
      VSAVE(9,INJB)=FLOAT(NPARTS)+.1
      VSAVE(10,INJB)=1.
      AL1=AL1R*180./PI
      BE1=BE1R*180./PI
C
      IF(INJB.LT.100) GO TO 370
      WRITE(JV) INJB,((VSAVE(K,J),K=1,10),J=1,INJB)
      INJB=0
C
  370 CONTINUE
C$$   WRITE(M,*) " ATOBTOC =",NTOP,NSID,NBOT
      IF(INJB.GT.0) WRITE(JV) INJB,((VSAVE(I,J),
     1 I=1,10),J=1,INJB)
      INJB=0
      IF(INIT.EQ.1) RETURN
      JV=3-JV
      IV=3-JV
C
      IF(IPRINT.GT.8) WRITE(M,3410) IT,TIME,PARTCL
      IF(IPRINT.GT.3) WRITE(M,3415) DELTA,DTSEC
C
      DO 820 I=1,NTOT
      KOUNT(I)=0
      XKOUN(I)=0.
  820 CONTINUE
      IOPAR=1HE
      IF(IPART.EQ.1) IOPAR=1HI
      REWIND JV
      REWIND IV
  890 CONTINUE
      READ(IV) INJA,((VSAVE(I,J),I=1,10),J=1,INJA)
      IF(EOF(IV)) 820,870
  870 CONTINUE
      DO 910 INJ=1,INJA
      NPART=VSAVE(9,INJ)+.1
      IF(IPART.EQ.1.AND.VSAVE(8,INJ).LT.0.) GO TO 860
      IF(IPART.EQ.2.AND.VSAVE(8,INJ).GT.0.) GO TO 860
      NCACT=NCACT+1
C
C  NEW = 1 IMPLIES NEWLY INJECTED PARTICLE
C  FAYT = 0., -1., -2., IMPLIES NORMAL, ESCAPING, OR ABSORBED
```

```
C
      NEW=0
      FAYT=0.
      FATE=1H
      ZOLD=VSAVE(3,INJ)
      X=VSAVE(1,INJ)
      Y=VSAVE(2,INJ)
      Z=VSAVE(3,INJ)
      R=SQRT(X*X+Y*Y)
      XDOT=VSAVE(4,INJ)
      YDOT=VSAVE(5,INJ)
      ZDOT=VSAVE(6,INJ)
      DT=VSAVE(7,INJ)*CZMIN
      INT=0
      IF(INTER.EQ.0) CALL INTERP(R,Z,JG,IG,PHIC,RR,ZZ,NR,NZ,INT)
      IF(INTER.GT.0) CALL INTERP(R,Z,JG,IG,PHIC,RI,ZCI,IR,IZ,INT)
      E=PHI+SPEED*SPEED
      IF(VSAVE(7,INJ).LT.DELTA) NEW=1
      VEL=SQRT(XDOT*XDOT+YDOT*YDOT+ZDOT*ZDOT)
      IF(IPRINT.GT.3.AND.NEW.EQ.1) WRITE(M,2100) NPART,IDPAR,IG,JG,
     1 N1,U(N1),X,Y,Z,R,XDOT,YDOT,ZDOT,FATE,VEL
      IF(VSAVE(7,INJ).LT.DELTA) VEL=VSAVE(7,INJ)
C
      VELSQ=1.
      IF(NEW.EQ.0) GO TO 450
      PHIA=PHI*SPEED**2
      IF(X.EQ.RR(NR)) GO TO 420
      VELSQ=ZDOT**2-PHIA
      IF(VELSQ.GT.0.) ZDOT=SIGN(SQRT(VELSQ),ZDOT)
      IF(VELSQ.LE.0.) ZDOT=-ZDOT
      GO TO 430
  420 CONTINUE
      VELSQ=XDOT**2-PHIA
      IF(VELSQ.GT.0.) XDOT=SIGN(SQRT(VELSQ),XDOT)
      IF(VELSQ.LE.0.) XDOT=-XDOT
  430 CONTINUE
      IF(IPRINT.GT.3) WRITE(M,2100) NPART,IDPAR,IG,JG,
     1 N1,U(N1),X,Y,Z,R,XDOT,YDOT,ZDOT,FATE,VEL
      IF(VELSQ.LE.0.) GO TO 500
  450 CONTINUE
      IF(MODE.NE.1) GO TO 440
C
C  DISTRIBUTE FORCES INTO PHIR AND PHIZ
C
      CALL CLOUD(R,Z,PHIC,IFDSK,WT,WCLOUD)
      PHIR=0.
      PHIZ=0.
      N=0
      DO 460 I=1,NZ
      DO 460 J=1,NR
      N=N+1
```

```
      PHIR=PHIR+WT(N)*PHYR(N)
C
C  SIFT FOR SPECIAL HALF BOX MARKED BY IFDSK = -1
C
      IF(I.NE.NZA.OR.J.GT.JEDGE) GO TO 470
      IF(IFDSK(J).NE.-1) GO TO 470
C
      PHIZ=PHIZ+WT(N)*PHYALT(J)
      GO TO 460
C
  470 CONTINUE
      PHIZ=PHIZ+WT(N)*PHYZ(N)
  460 CONTINUE
C
  440 CONTINUE
      IF(MODE.NE.2) GO TO 465
C
C  CALCULATE WEIGHTS
C
      IF(NEW.NE.1.OR.PHI.GE.0..OR.IT.EQ.0) GO TO 465
      PHY=-PHI*TVOLTS/TVCHG
      XPHY=SQRT(PHY)
      ERFC=1.-ERF(XPHY,DENA)
      IF(XPHY.LT.2.) DENA=EXP(PHY)*ERFC
      IF(MONMAX.LE.1) VSAVE(10,INJ)=1./(1.-SQRT(PHY/(1.+PHY)))
      IF(MONMAX.EQ.2) VSAVE(10,INJ)=2./ROOTPI*XPHY*DENA
      VSAVE(10,INJ)=VSAVE(10,INJ)**.25
C
  465 CONTINUE
C
      XDOT=XDOT/SPEED
      YDOT=YDOT/SPEED
      ZDOT=ZDOT/SPEED
C
      CALL TRACK(R,Z)
C
      XDOT=XDOT*SPEED
      YDOT=YDOT*SPEED
      ZDOT=ZDOT*SPEED
      VSAVE(1,INJ)=X
      VSAVE(2,INJ)=Y
      VSAVE(3,INJ)=Z
      VSAVE(4,INJ)=XDOT
      VSAVE(5,INJ)=YDOT
      VSAVE(6,INJ)=ZDOT
      VSAVE(7,INJ)=DELTA
      R=SQRT(X*X+Y*Y)
      IF(R.GT.0.) RDOT=(X*XDOT + Y*YDOT)/R
      IF(R.EQ.0.) RDOT=SQRT(XDOT*XDOT+YDOT*YDOT)
C
C     CHECK WHETHER TRAJECTORY IS TERMINATED
```

```
C
      IF(R.GT.RR(NR).OR.Z.GT.ZZ(1)) GO TO 500
      IF(Z.LT.ZZ(NZ)) GO TO 500
C
      IF(Z*ZOLD.LT.0..AND.R.LT.RADIUS) GO TO 550
C
      IF(MODE.NE.1) GO TO 900
C
C  GENERATE CLOUD AND DISTRIBUTE CHARGE
      CALL CLOUD(R,Z,PHIC,IFDSK,WT,WCLOUD)
      DO 480 I=1,NTOT
      XKOUN(I)=XKOUN(I)+WT(I)
 480  CONTINUE
C
      GO TO 900
C
C     PARTICLE  ESCAPES
C
500   CONTINUE
C
      NCESC=NCESC+1
      NCACT=NCACT-1
      FATE=1HE
      FAYT=-1.
C
      GO TO 900
C
C     PARTICLE IS ABSORBED
C
 550  CONTINUE
C
      NCABS=NCABS+1
      NCACT=NCACT-1
      FATE=1HA
      FAYT=-2.
C
C
 900  CONTINUE
C
C     INTERPOLATE TO FIND IG,JG
      INT=2
      IF(FAYT.NE.-1.) CALL INTERP(R,Z,JG,IG,PHIG,RI,ZGI,IR,IZ,INT)
      N1=JG+NR*(IG-2)
C
C     DO SUMS FOR SUMMARY
C
      IF(FAYT.GE.0.) KOUNT(N1)=KOUNT(N1)+1
      IF(FAYT.GE.0..AND.MODE.NE.1) XKOUN(N1)=XKOUN(N1)+VSAVE(10,INJ)
      IF(IPART.EQ.2.AND.FAYT.EQ.-2.) NCHGE(JG)=NCHGE(JG)+1
      IF(IPART.EQ.1.AND.FAYT.EQ.-2.) NCHGI(JG)=NCHGI(JG)+1
C
```

```
C     WRITE SUMMARY FOR CURRENTLY ACTIVE PARTICLE
C       NEED TO ADD ROOT TO LISTINGS LATER
C
      IF(IPRINT.GT.3.AND.NEW.EQ.0) WRITE(M,2100) NPART,IOPAR,
     1 IG,JG,N1,U(N1),X,Y,Z,R,XDOT,YDOT,ZDOT,FATE,VEL
      IF(IPRINT.GT.3.AND.NEW.EQ.1) WRITE(M,2200) NPART,IOPAR,
     1 IG,JG,N1,U(N1),X,Y,Z,R,XDOT,YDOT,ZDOT,FATE,VEL
C
C     SAVE CURRENTLY ACTIVE PARTICLES
C
      IF(FAYT.LT.0.) GO TO 910
  860 CONTINUE
      INJB=INJB+1
      IF(INJ.EQ.INJB) GO TO 910
      DO 915 I=1,9
      VSAVE(I,INJB)=VSAVE(I,INJ)
  915 CONTINUE
C
  910 CONTINUE
C
      IF(INJB.GT.0) WRITE(JV) INJB,((VSAVE(I,J),I=1,10),J=1,INJB)
      INJB=0
      GO TO 890
C
  880 CONTINUE
C
      IF(IPRINT.EQ.0) RETURN
C
      WRITE(M,3420) PARTCL,(BLK,I,I=1,JEDGE)
      IF(IPART.EQ.2) WRITE(M,3430) NZA,(NCHGE(I),
     1 I=1,JEDGE)
      IF(IPART.EQ.1) WRITE(M,3430) NZA,(NCHGI(I),
     1 I=1,JEDGE)
C
      N1=0
      N2=0
      WRITE(M,3440) PARTCL,(BLK,I,I=1,NR)
      DO 920 I=1,NZ
      N1=N2+1
      N2=N2+NR
      WRITE(M,3430) I,(KOUNT(J),J=N1,N2)
  920 CONTINUE
C
      RETURN
C
 1000 FORMAT(1H1/21H FLUXES AND DENSITIES,4X,4HIT =,I3,4X,2A5,
     1 13H WITH SPEED =,1PE15.4,7H CM/SEC,4X,22H AND CURRENT DENSITY =,
     2 E15.4,14H PICOAMP/CM**2)
 1010 FORMAT(/5X,41HDIMENSIONLESS POTENTIAL ARRAY - ABOVE Z=0
     1 5X,32H(IN UNITS OF MINUS TVIONS/XMASS)
     2 //1X,3HR =,14F9.3/ (/4X,14F9.3))
```

```
 1020 FORMAT(/5X,41HDIMENSIONLESS POTENTIAL ARRAY - ABOVE Z=0
     1 5X,20H(IN UNITS OF TVIONS)
     2 //1X,3HR =,14F9.3/ (/4X,14F9.3))
 1100 FORMAT(/5H LINE,I4,5X,2HZ=,F7.3/(/7F16.8))
 1200 FORMAT(///1X, 41HDIMENSIONLESS POTENTIAL ARRAY - BELOW Z=0 )
 1300 FORMAT(1X,17HSINGLE TRAJECTORY/1X,23HANGLES ALPHA AND BETA =,
     12F15.8,11H DEGREES  =,2F15.8,8H RADIANS/9H ENERGY =,F15.8,5HVOLTS)
 2000 FORMAT(1X,I5,1P11E10.3,2I5)
 2100 FORMAT(1X,I5,1H-,A1,3I3,1P8E11.3,2X,A1,E11.3)
 2200 FORMAT(1X,I5,1H-,A1,3I3,1P8E11.3,2X,A1,6H  RAN=,E10.3)
 2700 FORMAT(/1X,2A6,5X,16HENERGY CHANGES =, 1P2E10.2,
     1 28H (WHOLE ORBIT, MAX PER STEP) )
 3400 FORMAT(/5X,33HABSORPTION SUMMARY FOR IT CYCLE =,I5,
     1 2X,9HAT TIME =,1PE10.3,2X,2A5,2X,5X,23HTOTAL NUMBER ABSORBED =,
     2 I7/8X,27HNUMBER ABSORBED THIS STEP =,I5,8X,15HNUMBER ESCAPING,
     3 12H THIS STEP =,I5,8X,25HNUMBER CURRENTLY ACTIVE =,I5)
 3410 FORMAT(1H1//,5X,38HPOPULATION SUMMARY FOR ITERATION  IT =,
     1 I5,5X,9HAT TIME =,1PE10.3,2X,2A5)
 3415 FORMAT(5X,6HDELTA=,1PE11.3,5X,11H(MEANS DT =,E11.3,5H SEC)
     1 //10X,7HI   J   N,2X,9HPOTENTIAL,6X,1HX,10X,1HY,10X,1HZ,10X,1HR,
     2 8X,42HXDOT          YDOT         ZDOT         VELOCITY,
     3 /20X,5HVOLTS,9X,2HCM,9X,2HCM,9X,2HCM,9X,2HCM,7X,6HCM/SEC,
     4 6X,6HCM/SEC,4X,6HCM/SEC,6X,6HCM/SEC)
 3420 FORMAT(//5X,31HSUMMARY OF ABSORBED CHARGES FOR,2X,2A5//12X,9(A2,
     1 2HR(,I1,1H)),11(A1,2HR(,I2,1H)))
 3430 FORMAT(/5X,2HZ(,I2,1H),20I6)
 3440 FORMAT(//5X,34HSUMMARY OF SPACE CHARGE MATRIX FOR,2X,2A5//12X,9(A2
     1 ,2HR(,I1,1H)),11(A1,2HR(,I2,1H)))
      END
```

```
      SUBROUTINE CLOUD(RCD,ZCD,PHIC,IFDSK,WT,WCLOUD)
C***********************************************************************
C*****
C  DISTRIBUTION OF FINITE-CLOUD PARTICLE AMONG GRID BOXES
C     (CHARGES AND FORCES)
C
C  VERSION OF 15 JAN 79.  LAST ALTERATION 2 MAR 79.....
C*****
C***********************************************************************
      DIMENSION PHIC(51),WT(51),IFDSK(11)
      COMMON N,IV,IFIRST,JFIRST,JEDGE,NR,NZA,NZ,NTOT,PI,IT,IR,IZ,
     1 NONMAX,TIME,IPRINT,RADIUS,RR(11),ZZ(11),U(51),Q(51),
     2 AREAH(11),DELZ(11),RI(11),ZCI(11)
      NIJ(I,J)=NR*(I-2)+J
      NZAP=NZA+1
C
C  DETERMINE R AND Z COORDINATES OF CORNERS OF CLOUD
C
      ZH=AMIN1(ZCD+WCLOUD,ZCI(1))
      ZL=AMAX1(ZCD-WCLOUD,ZCI(IZ))
      RH=AMIN1(RCD+WCLOUD,RI(IR))
      RL=AMAX1(RCD-WCLOUD,0.)
C
C  DETERMINE IG AND JG CORNERS OF GRID CONTAINED WITHIN CLOUD
C
      CALL INTERP(RL,ZL,JGL,IGL,PHIC,RI,ZCI,IR,IZ,2)
      CALL INTERP(RH,ZH,JGH,IGH,PHIC,RI,ZCI,IR,IZ,2)
C
C  DETERMINE VOLUME WEIGHTS OF EACH IG JG BOX CONTAINED IN CLOUD
C
      DO 80 II=1,NTOT
      WT(II)=0.
   80 CONTINUE
C
      DO 90 II=1,NR
      IFDSK(II)=0
   90 CONTINUE
C
      DO 10 I=IGH,IGL
      DO 20 J=JGL,JGH
      N1=NIJ(I,J)
      W2=AREAH(J)
      IF(I.LE.NZA) W1=DELZ(I-1)
      IF(I.EQ.NZAP) W1=DELZ(NZA)+DELZ(NZAP)
      IF(I.GT.NZAP) W1=DELZ(I)
      IF(I.EQ.IGH) W1=ZH-ZCI(IGH)
      IF(I.EQ.IGL) W1=ZCI(IGL-1)-ZL
      IF(IGH.EQ.IGL) W1=2.*WCLOUD
C
      IF(J.EQ.JGL) W2=PI*(RI(JGL+1)**2-RL**2)
      IF(J.EQ.JGH) W2=PI*(RH**2-RI(JGH)**2)
```

```
      IF(JGH.EQ.JGL) W2=2.*WCLOUD
C
      IF(ZL.GT.0..OR.ZH.LT.0..OR.J.GT.JEDGE) GO TO 30
C
      IF(ZCO.LT.0..AND.I.LE.NZAP) GO TO 40
      IF(ZCO.GE.0..AND.I.GE.NZAP) GO TO 50
      GO TO 30
C
C  BOX IS IN SHADOW
C
 50   CONTINUE
C  BELOW DISC
      W1=0.
      IF(I.EQ.NZAP) W1=AMIN1(DELZ(NZA),ZH)
      GO TC 30
C
 40   CONTINUE
C  ABOVE DISC
      W1=0.
      IF(I.NE.NZAP) GO TO 30
      IFDSK(J)=-1
      W1=AMIN1(DELZ(NZAP),-ZL)
C
 30   CONTINUE
      IF(I.EQ.NZAP.AND.ZH.LT.0.) IFDSK(J)=-1
      WT(N1)=W1*W2
 20   CONTINUE
 10   CONTINUE
C
C  COMPLETE WEIGHTS
C
      WVOL=0.
      DO 60 I=1,NTCT
      WVOL=WVOL+WT(I)
 60   CONTINUE
      DO 70 I=1,NTCT
      WT(I)=WT(I)/WVOL
 70   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ABNOR(ERR)
C
C*****
C
C  SUBROUTINE ABNOR IS A DUMMY ROUTINE TO FORCE A MODE 2 EXIT
C     WHEN ANY NON STANDARD STOP CONDITION IS DETECTED
C
C*****
C
      PRINT 3001,ERR
 3001 FORMAT(1X,*ABNORMAL STOP REQUESTED*,5X,A10)
      A=0.
      C=B/A
      RETURN
      END
*EOR
  MAXWELLIAN ISOTROPIC CASE    MARCH 1979
    2     5     3     3     0     1
0.         40.        60.        80.        100.
50.        25.        0.
0.         -25.       -50.
-10.       -10.
    0    1    0    40    20    3    20    25
.00803     0.         .1         .1         75000.     15.
 *EOR
  MAXWELLIAN ISOTROPIC CASE    OCTOBER 1978
   24   50   26   26    0    0
0.         3.         6.         9.         12.        14.        16.        18
20.        22.        24.        26.        28.        30.        32.        34
36.        38.        40.        42.        44.        46.        48.        49
50.5       52.        54.        56.        58.        60.        62.        64
66.        68.        70.        72.        74.        76.        78.        80
82.        84.        86.        88.        90.        92.        94.        96
98.        100.
50.        48.        46.        44.        42.        40.        38.        36
34.        32.        30.        28.        26.        24.        22.        20
18.        16.        14.        12.        10.        8.         6.         4.
2.         0.
0.         -2.        -4.        -6.        -8.        -10.       -12.       -1
-16.       -18.       -20.       -22.       -24.       -26.       -28.       -3
-32.       -34.       -36.       -38.       -40.       -42.       -44.       -4
-48.       -50.
-10.       -10.       -10.       -10.       -10.       -10.       -10.       -1
-10.       -10.       -10.       -10.       -10.       -10.       -10.       -1
-10.       -10.       -10.       -10.       -10.       -10.       -10.       -1
    0    1    0    10    20    3    20    25
.1         0.         .1         .1         75000.     15.
```

```
      PROGRAM SUMRTE(TAPE1,INPUT,OUTPUT,TAPE2,TAPE6=OUTPUT,TAPE5=INPUT)
      DIMENSION
     1 KHABSI(12),KHACTI(12),MESG(5),OSET(4),
     2 KHABSE(12),KHACTE(12),LINE(100),PVEC(4),IPSYM(4),PSCL(4)
      NAMELIST /CONTL/ DENCC,AMN,AMX,BMN,BMX,CMN,CMX,DMN,DMX
C
C   DEFAULT CONTL ASSIGNMENTS
C
      DENCC=10000.
      AMN=0.
      AMX=30.
      BMN=2500.
      BMX=5000.
      CMN=2500.
      CMX=5000.
      DMN=-1.
      DMX=4.
C
      IPSYM(1)=1HA
      IPSYM(2)=1HB
      IPSYM(3)=1HC
      IPSYM(4)=1HD
C
C   READ CONTL PARAMETERS
C
      CALL CONNEC(5)
      CALL CONNEC(6)
      WRITE(6,2002)
 2002 FORMAT(1X,*ENTER DENCC AND RANGES OF PRINTER PLOT VARIABLES*,
     +* IN NAMELIST FORMAT*/)
      READ(5,CONTL)
      WRITE(6,2004)
 2004 FORMAT(1X,*ENTER THIRTY CHARACTER IDENTIFIER*/)
      READ(5,1001) MESG
 1001 FORMAT(5A10)
      PSCL(1)=AMX-AMN
      PSCL(2)=BMX-BMN
      PSCL(3)=CMX-CMN
      PSCL(4)=DMX-DMN
      OSET(1)=AMN
      OSET(2)=BMN
      OSET(3)=CMN
      OSET(4)=DMN
      CALL DISCON(6)
      REWIND 6
      REWIND 1
      REWIND 2
C*****
 1000 CONTINUE
      JLINE=0
C
```

```
C  INITIALIZE MATRICES FOR ANALYSIS OF VARIANCES
C
      DO 620 I=1,12
      KHABSI(I)=0
      KHACTI(I)=0
      KHABSE(I)=0
      KHACTE(I)=0
  620 CONTINUE
  650 CONTINUE
      READ(1,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,
     1 RMIABS,REMAES,RMIACT,RMEACT,AVIABS,AVEABS,AVIACT,AVEACT,XMAX
      JLINE=JLINE+1
      IF(MOD(JLINE,50).EQ.1) WRITE(6,2001) DENCC
 2001 FORMAT(1H1,5X,*TABULATED STATISTICAL SUMMARY OF DATA AS A *
     +*FUNCTION OF TIME.  DENCC = *,1PE12.4)
      IF(MOD(JLINE,50).EQ.1) WRITE(6,2005) MESG
 2005 FORMAT(10X,5A10)
      IF(MOD(JLINE,50).EQ.1) WRITE(6,3002)
      JVAR=MOD(JLINE,10)+1
      IF(EOF(1)) 99,660
  660 CONTINUE
C
C  ADD TO REMAINING VARIANCE SUMS
C
      KHABSI(11)=KHABSI(11)+NIABS-KHABSI(JVAR)
      KHACTI(11)=KHACTI(11)+NIACT-KHACTI(JVAR)
      KHABSI(12)=KHABSI(12)+NIABS*NIABS-KHABSI(JVAR)*KHABSI(JVAR)
      KHACTI(12)=KHACTI(12)+NIACT*NIACT-KHACTI(JVAR)*KHACTI(JVAR)
      KHABSI(JVAR)=NIABS
      KHACTI(JVAR)=NIACT
      KHABSE(11)=KHABSE(11)+NEABS-KHABSE(JVAR)
      KHACTE(11)=KHACTE(11)+NEACT-KHACTE(JVAR)
      KHABSE(12)=KHABSE(12)+NEABS*NEABS-KHABSE(JVAR)*KHABSE(JVAR)
      KHACTE(12)=KHACTE(12)+NEACT*NEACT-KHACTE(JVAR)*KHACTE(JVAR)
      KHABSE(JVAR)=NEABS
      KHACTE(JVAR)=NEACT
  730 CONTINUE
C
C  BEGIN CALCULATION OF VARIANCES
C
      DELT=MIN0(JLINE,10)
      AVIABS=FLOAT(KHABSI(11))/DELT
      AVIACT=FLOAT(KHACTI(11))/DELT
      RMIABS=SQRT(FLOAT(KHABSI(12))/DELT-AVIABS*AVIABS)
      RMIACT=SQRT(FLOAT(KHACTI(12))/DELT-AVIACT*AVIACT)
      AVEABS=FLOAT(KHABSE(11))/DELT
      AVEACT=FLOAT(KHACTE(11))/DELT
      RMEABS=SQRT(FLOAT(KHABSE(12))/DELT-AVEABS*AVEABS)
      RMEACT=SQRT(FLOAT(KHACTE(12))/DELT-AVEACT*AVEACT)
 3002 FORMAT(///25X,21HSUMMARY FOR THIS STEP,16X,19HTEN-STEP RMS ERRORS,
     1 17X,17HTEN-STEP AVERAGES/22X,4HIONS,12X,9HELECTRONS,1X,2(3X,2(7X,
```

```
      2 1HI,7X,1HE))/1X,5HCYCLE,3X,4HTIME,2(4X,14HABS  ESC   ACT),
      3 2(8X,3HABS,5X,3HABS,5X,3HACT,5X,3HACT),2X,4HPMAX)
        WRITE(6,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,
      1 RMIABS,RMEABS,RMIACT,RMEACT,AVIABS,AVEABS,AVIACT,AVEACT,XMAX
        WRITE(2,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,
      1 RMIABS,RMEABS,RMIACT,RMEACT,AVIABS,AVEABS,AVIACT,AVEACT,XMAX
3001    FORMAT(1X,I4,1PE10.3,1X,I3,2I6,3X,I3,2I6,3X,0P4F8.1,3X,4F8.1,F8.3)
        GO TO 650
99      CONTINUE
        REWIND 2
        WRITE(6,2003) DENCC
2003    FORMAT(1H1,5X,*PRINTER PLOT SUMMARY OF DATA AS A FUNCTION*
      +* OF TIME.   DENCC = *,1PE12.4)
        WRITE(6,2005) MESG
        WRITE(6,2006) AMN,AMX,BMN,BMX,CMN,CMX,DMN,DMX
2006    FORMAT(/5X,*SYMBOL CORRESPONDENCE*/10X,*A  =  ION ABSORPTIONS*,
      +13X,*SCALED FROM*,5X,F8.1,*  TO*,F8.1/
      +10X,*B  =  ION POPULATION*,14X,*SCALED FROM*,5X,F8.1,*  TO*,F8.1/
      +10X,*C  =  ELECTRON POPULATION*,9X,*SCALED FROM*,5X,F8.1,*  TO*,
      +F8.1/10X,*D  =  MAXIMUM POTENTIAL*,11X,*SCALED FROM*,5X,F8.1,
      +*  TO*,F8.1//)
30      CONTINUE
        READ(2,3001) IT,TIME,NIABS,NIESC,NIACT,NEABS,NEESC,NEACT,RMIABS,
      1 RMEABS,RMIACT,RMEACT,PVEC(1),AVEABS,PVEC(2),PVEC(3),PVEC(4)
        IF(EOF(2)) 40,50
50      CONTINUE
        IFLG=0
        IF(MOD(IT,10).EQ.0) IFLG=1
        DO 10 II=1,100
        LINE(II)=1H
        IF(IFLG.EQ.1.AND.MOD(II,10).EQ.0) LINE(II)=1H+
10      CONTINUE
        DO 20 II=1,4
        IPV=(PVEC(II)-OSET(II))/PSCL(II)*100.+1.
        IPV=MIN0(MAX0(IPV,1),100)
        LINE(IPV)=IPSYM(II)
20      CONTINUE
        IF(IFLG.EQ.1) WRITE(6,3004) IT,LINE
        IF(IFLG.NE.1) WRITE(6,3003) LINE
3003    FORMAT(10X,1H*,100A1,1H*)
3004    FORMAT(4X,I3,3X,1H*,100A1,1H*)
        GO TO 30
40      CONTINUE
        END
```